

68

MICRO JOURNAL

Australia A \$ 4.75 New Zealand NZ \$ 6.50
 Singapore S \$ 9.45 Hong Kong H \$23.50
 Malaysia M \$ 9.45 Sweden 30-SEK

\$2.95 USA

Motorola Microprocessors:

Apple Macintosh™ 68000
 GIMIX™,SSB™,SWTPC™ 6809
 Tandy CoCo™ 6809

Advanced Operating Systems:

OS-9™,UniFLEX™,FLEX™
 STAR-DOS™,Mac-DOS

Programming Languages:

C, Assembler, K-BASIC
 Compilers, Dissassemblers

VOLUME VII ISSUE V • Devoted to the 68XX User • May 1985

This Month:

Ada Programming Language
 CoCo Using FLEX/STAR-DOS
 OS-9, C User Applications

SERVING THE 68XX USER WORLDWIDE

VCC 1
 NMIO 2
 FIO 3
 FIO 4
 BS 5
 BA 6
 VCC 7
 A0 8
 A1 9
 A2 10
 A3 11
 A4 12
 A5 13
 A6 14
 A7 15
 A8 16
 A9 17
 A10 18
 A11 19
 A12 20

MC6809E CPU

DIO 1
 GIO 2
 GIO 3
 GIO 4
 DO 5
 AS 6
 UD 7
 DS 8
 DS 9
 DS 10
 DS 11
 DS 12
 DS 13
 DS 14
 DS 15
 DS 16
 DS 17
 DS 18
 DS 19
 DS 20
 DS 21
 DS 22
 DS 23
 DS 24
 DS 25
 DS 26
 DS 27
 DS 28
 DS 29
 DS 30
 DS 31
 DS 32

MC6800 CPU

VCC 1
 NMIO 2
 FIO 3
 FIO 4
 BS 5
 BA 6
 VCC 7
 A0 8
 A1 9
 A2 10
 A3 11
 A4 12
 A5 13
 A6 14
 A7 15
 A8 16
 A9 17
 A10 18
 A11 19
 A12 20

MC6809 CPU



WE DON'T PLAY GAMES



X-12+ A SERIOUS COMPUTER IN A DESKTOP PACKAGE

Multiprocessor Technology - Combination of 8, 16 and 32 bit types

1.0 Megabyte Memory - Insures no limitation on programs

"Winchester" Disk System - Fast response, large storage capacity

UniFlex^{*} Operating System - The standard of comparison

Hardware Floating Point - Unmatched speed in a small system

Up to Three Terminals - Instant expansion

* Trademark of Technical Systems Consultants



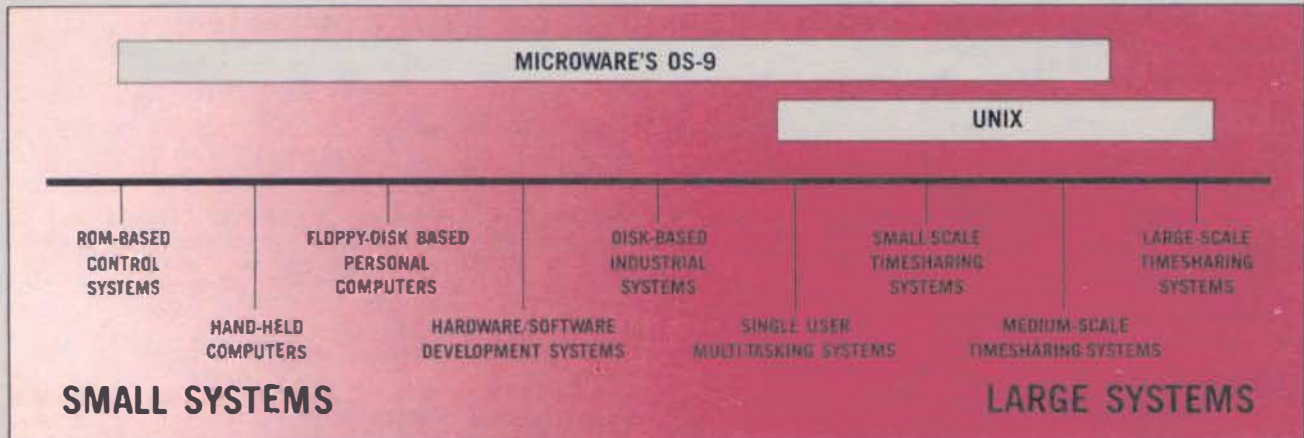
SOUTHWEST TECHNICAL PRODUCTS CORPORATION

219 W. RHAPSODY

SAN ANTONIO, TEXAS 78216

(512) 344-0241

Only Microware's OS-9 Operating System Covers the Entire 68000 Spectrum



Is complicated software and expensive hardware keeping you back from Unix? Look into OS-9, the operating system from Microware that gives 68000 systems a Unix-style environment with much less overhead and complexity.

OS-9 is versatile, inexpensive, and delivers outstanding performance on any size system. The OS-9 executive is much smaller and far more efficient than Unix because it's written in fast, compact assembly language, making it ideal for critical real-time applications. OS-9 can run on a broad range of 8 to 32 bit systems based on the 68000 or 6809 family MPUs from ROM-based industrial controllers up to large multiuser systems.

OS-9'S OUTSTANDING C COMPILER IS YOUR BRIDGE TO UNIX

Microware's C compiler technology is another OS-9 advantage. The compiler produces extremely fast, compact, and ROMable code. You can easily develop and port system or application software back and forth to standard Unix systems. Cross-compiler versions for

VAX and PDP-11 make coordinated Unix/OS-9 software development a pleasure.

SUPPORT FOR MODULAR SOFTWARE — AN OS-9 EXCLUSIVE

Comprehensive support for modular software puts OS-9 a generation ahead of other operating systems. It multiplies programmer productivity and memory efficiency. Applica-

tion software can be built from individually testable software modules including standard "library" modules. The modular structure lets you customize and reconfigure OS-9 for specific hardware easily and quickly.

A SYSTEM WITH A PROVEN TRACK RECORD

Once an underground classic, OS-9 is now a solid hit. Since 1980 OS-9 has been ported to over a hundred 6809 and 68000

systems under license to some of the biggest names in the business. OS-9 has been imbedded in numerous consumer, industrial, and OEM products, and is supported by many independent software suppliers.

Key OS-9 Features At A Glance

- Compact (16K) ROMable executive written in assembly language
- User "shell" and complete utility set written in C
- C-source code level compatibility with Unix
- Full Multitasking/multiuser capabilities
- Modular design - extremely easy to adapt, modify, or expand
- Unix-type tree structured file system
- Rugged "crash-proof" file structure with record locking
- Works well with floppy disk or ROM-based systems
- Uses hardware or software memory management
- High performance C, Pascal, Basic and Cobol compilers

Microware
OS-9™

MICROWARE SYSTEMS CORPORATION
1866 NW 114th Street
Des Moines, Iowa 50322
Phone 515-224-1929
Telex 910-520-2535

Microware Japan, Ltd
3-8-9 Baraki, Ichikawa City
Chiba 272-01, Japan
Phone 0473(28)4493
Telex 299-3122

OS-9 is a trademark of Microware and Motorola. Unix is a trademark of Bell Labs.

'68'

Portions of the text for '68' Micro Journal were prepared using the following furnished Hard/Software:

COMPUTERS - HARDWARE

Southwest Technical Products
219 W. Rhapsody
San Antonio, TX 78216
OS9 - 5/8 DMF Disk - CDS1 - 8212W - Sprint 3 Printer

GIMIX Inc.
1337 West 37th Place
Chicago, IL 60609
Super Mainframe - OS9 - FLEX - Assorted Hardware

EDITORS - WORD PROCESSORS

Technical Systems Consultants, Inc.
111 Providence Road
Chapel Hill, NC 27514
FLEX - Editor - Text Processor

Great Plains Computer Co., Inc.
PO Box 916
Idaho Falls, ID 83401
Stylograph - Mail Merge - Spell

Editorial Staff

Don Williams Sr.	Publisher
Larry E. Williams	Executive Editor
Tom E. Williams	Production Editor
Robert L. Nay	Technical Editor

Administrative Staff

Mary Robertson	Office Manager
Penny Williams	Subscriptions
Christine Kocher	Accounting

Contributing Editors

Ron Anderson	Norm Connors
Peter Dibble	William E. Fisher
Dr. Theo Elbert	Carl Mann
Dr. E. M. Pass	Ron Voigts
Philip Lucido	

Special Technical Projects

Clay Abrams K6AEP
Tom Hunt

CONTENTS

Vol. VII, Issue V

May 85

FLEX USER Notes.....	7	Anderson
OS9 USER Notes.....	11	Dibble
C USER Notes.....	12	Pass
68000 USER Notes.....	18	Lucido
Whimsical Revisited.....	20	Anderson
ADA ^R And The 68000.....	22	Elbert
Basic OS-9.....	26	Voigts
CoCo USER Notes.....	29	Mann
Using FLEX/Star DOS.....	32	Brumley
The Apple Macintosh.....	33	Nay
Ramblings.....	41	Glennon, DMW Casneuf
Bit Bucket.....	47	
Classified Advertising.....	59	

MICRO JOURNAL

Send All Correspondence To:

Computer Publishing Center
68' Micro Journal
5900 Cassandra Smith Rd.
Mixon, Tn. 37343

Phone (615) 842-4600 or Telex 558 414 PVT BTH

Copyrighted 1985 by Computer Publishing Inc.

68' Micro Journal is published 12 times a year by Computer Publishing Inc. Second Class Postage Paid ISSN 0194-5025 at Mixson, Tn. and additional entries. Postmaster: send form 3597 to 68' Micro Journal, POB 849 Mixson, Tn. 37343.

Subscription Rates

1 Year \$24.50 U.S.A., Canada & Mexico Add \$9.50 a Year. Other Foreign Add \$12 a Year for Surface, Airmail Add \$48 a Year. Must be in U.S. currency!

How or Articles For Publication

Articles submitted for publication should include authors name, address, telephone number and date. Articles should be on either 5 or 8 inch disk in STYLOGRAPH or TSC Editor format with 3.5 inch Disk width. All disks will be returned. Articles submitted on paper should be 4.5 inches in width (including Source Listings) for proper reductions. Please Use A Dark Ribbon!! No Blue Ink!!! Single space on 8X11 bond or better grade paper. No hand written articles accepted. Disks should be in FLEX2 6800 or FLEX9 6809 any version or OS-9 any version.

The following TSC Text Processor commands ONLY should be used: .sp space, .pp paragraph, .fl fill and .nf no fill. Also please do not format within the text with multiple spaces. We will enter the rest at time of editing.

All STYLOGRAPH commands are acceptable except .pg page command. We print edited text files in continuous text form.

Letters To The Editor

All letters to the editor should comply with the above requirements and must be signed. Letters of "gripes" as well as "praise" are solicited. We reserve the right to reject any submission for lack of "good taste" and we reserve the right to define "good taste".

Advertising Rates

Commercial advertisers please contact 68' Micro Journal advertising department for current rate sheet and requirements.

Classified Advertising

All classified ads must be non-commercial. Minimum of \$9.50 for first 20 words and .45 per word after 20. All classifieds must be paid in advance. No classified ads accepted over the phone.

GIMIX HAS THE 6809 SYSTEM TO SUIT YOUR NEEDS

HARDWARE

All systems feature the **GIMIX CLASSY CHASSIS**: with a ferro-resonant constant voltage power supply, gold plated bus connectors, and plenty of capacity for future expansion.

Static RAM and double-density DMA floppy disk controllers are used exclusively in all systems.

All systems are guaranteed for 2 MHz operation and include complete hardware and software documentation, necessary cables, filler plates, etc.

Systems are assembled using burned-in and tested boards, and all disk drives are tested and aligned by **GIMIX**.

You can add additional components to any system when ordering, or expand it in the future by adding RAM, I/O, etc.

GIMIX lets you choose from a wide variety of options to customize your system to your needs.

OS-9 GMX III/FLEX SYSTEMS (#79)

The #79 super system now includes (in addition to the above): the **GMX 6809 CPU III**, a **256K CMOS Static RAM Board (#72)**, and a **3-port Intelligent Serial I/O Processor (#11)**.

The **GMX 6809 CPU III** can perform high-speed DMA transfers from memory to memory and uses memory attributes and illegal instruction trapping to protect the system and users from program crashes. If a user program crashes, only that user is affected; other users are unaware of the problem.

The **3-Port Intelligent Serial I/O Board (#11)** significantly reduces system overhead by handling routine I/O functions; freeing the host CPU for running user programs. This improves overall system performance and allows user terminals to be run at up to 19.2K baud.

with dual 40 track DSDD drives	\$5998.79
with dual 80 track DSDD drives	\$6198.79
with #88 dual 8" DSDD drive system	\$7698.79
with #90 19MB Winchester subsystem and one 80 track	\$8898.79
with a 47MB Winchester subsystem and one 80 track	\$10,898.79
with a 47MB plus a 6MB removable pack Winchester subsystem and one 80 track drive	\$12,398.79

TO ORDER BY MAIL: SEND CHECK OR MONEY ORDER OR USE YOUR VISA OR MASTER CHARGE. Please allow 3 weeks for personal checks to clear. U.S. orders add \$5 handling if order is under \$200.00. Foreign orders add \$10 handling if order is under \$200.00. Foreign orders over \$200.00 will be shipped via Emery Air Freight COLLECT, and we will charge no handling. All orders must be prepaid in U.S. funds. Please note that foreign checks have been taking about 8 weeks for collection so we would advise wiring money, or checks drawn on a bank account in the U.S. Our bank is the Continental Illinois National Bank of Chicago, 231 S. LaSalle Street, Chicago, IL 60603, account #73-32033.

SOFTWARE

All **OS-9/FLEX** systems allow you to software select either operating system.

Also included is the **GMXBUG** monitor and, in systems with 128K or more of RAM, **GMX-VDISK** for **FLEX**.

All **GIMIX OS-9** systems include **Microware's Editor, Assembler, Debugger, Basic09,** and **Runb**; and the **GMX** versions of **RMS** and **DO** for **OS-9**.

All **GIMIX** versions of **OS-9** can read and write **RS color computer format OS-9** disks, as well as the **Microware/GiMIX standard format**.

New and exclusive with **OS-9 GMX III** systems is the **GMX OS-9 Support ROM**, a monitor for **OS-9** that includes memory diagnostics and allows the system to boot directly from either hard disk or floppy.

A wide variety of languages and other software is available for use with either **OS-9** or **FLEX**.

OS-9 GMX I / FLEX SYSTEMS #49

The #49 systems include 64KB static RAM, #05 CPU, #43 2 port serial board.

with dual 40 track DSDD drives	\$3998.49
with dual 80 track DSDD drives	\$4198.49
with #88 dual 8" DSDD drive system	\$5698.49
with #90 19MB Winchester subsystem and one 80 track	\$6898.49

OS-9 GMX II / FLEX SYSTEMS #39

The #39 systems include 128KB static RAM, #05 CPU, #43 2 port serial board.

with dual 40 track DSDD drives	\$4498.39
with dual 80 track DSDD drives	\$4698.39
with #88 dual 8" DSDD drive system	\$6198.39
with #90 19MB Winchester subsystem and one 80 track	\$7398.39

GIMIX DOES NOT GUARANTEE PERFORMANCE OF ANY GIMIX SYSTEMS, BOARDS OR SOFTWARE WHEN USED WITH OTHER MANUFACTURERS PRODUCT.

EXPORT MODELS: ADD \$30 FOR 50MHz. POWER SUPPLIES.

GIMIX, Inc. reserves the right to change pricing, terms, and products specifications at any time without further notice.

ALL PRICES ARE F.D.B. CHICAGO

Contact **GIMIX** for price and availability of **UniFLEX** and **UniFLEX GMXIII** Systems.

NOTE on all drive systems: Dual 40 track drives have about 700KB of formatted capacity; dual 80's about 1.400KB; dual 8" about 2.000KB. The formatted capacity of hard disks is about 80% of the total capacity.

Want to expand your system to a megabyte of Static RAM and 15 users?

Simply add additional memory and I/O boards. Your **GIMIX** system can grow with your needs. Contact us for a complete list of available boards and options.

#72 256KB CMOS STATIC RAM board	
with battery back up	\$1898.72
#64 64KB CMOS STATIC RAM board	
with battery back up	\$528.64
#67 64KB STATIC RAM board	\$478.67
#11 3 port intelligent serial I/O board	\$498.11
#43 2 port serial I/O board	\$128.43
#42 2 port parallel I/O board	\$88.42
#95 cable sets (1 needed per port), specify board	\$24.95

NOW SHIPPING !

UniFLEX
GMX III Systems

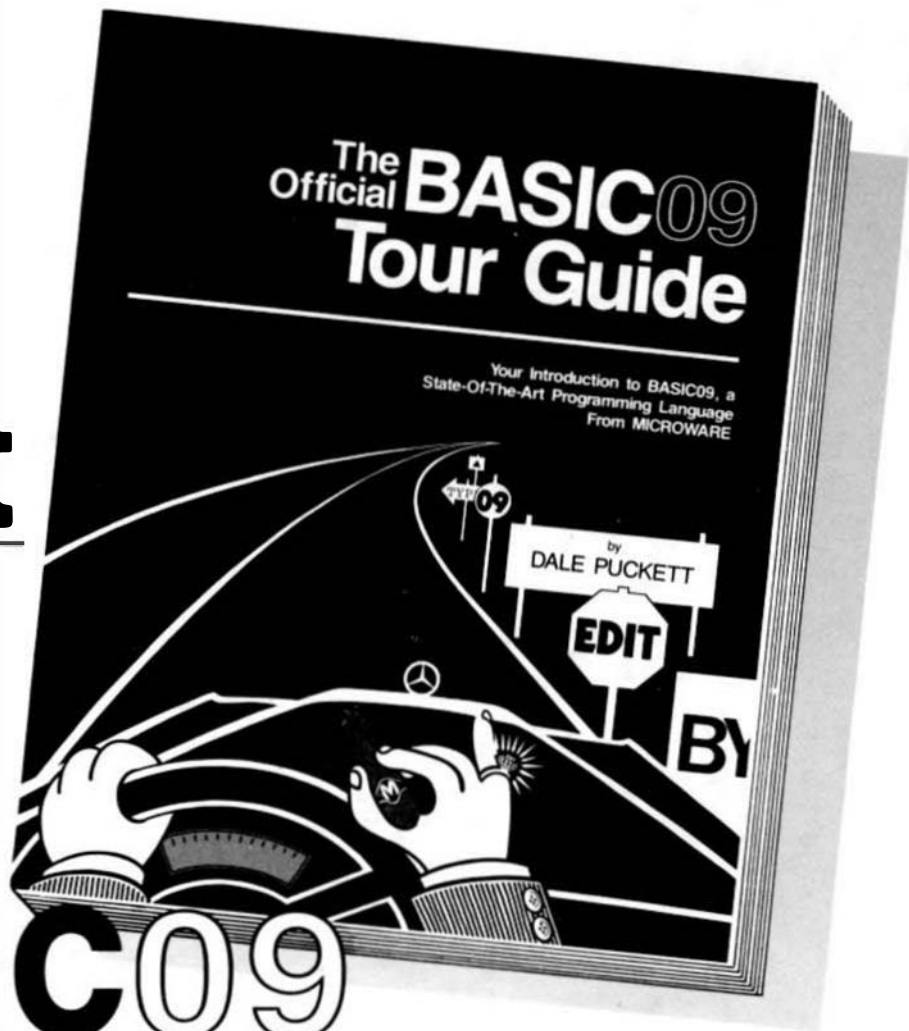
GIMIX inc.

1337 WEST 37th PLACE
CHICAGO, ILLINOIS 60609
(312) 927-5510 • TWX 910-221-4055



©1984 GIMIX, INC. 4-84

Get the most out of BASIC09



The **OFFICIAL BASIC09 TOUR GUIDE** is skillfully written in a friendly and easy-to-read style. Just perfect for those new to computers and to BASIC09. It's also a *valuable reference book* for programmers, engineers, students and hobbyists, providing an in-depth look at BASIC09 plus an overview of the OS-9 operating system. Comprehensive reference sections on BASIC09 and OS-9 commands are also included. The book "maps" out your route through the Mercedes of Basics... BASIC09 and puts you in the driver's seat in no time. Fasten your seatbelt, sit back and enjoy the ride to perfecting your programming skills.

MICROWARE . . .

The **OFFICIAL BASIC09 TOUR GUIDE** comes from the people who wrote BASIC09. As the leader in 6809 system software, we at MICROWARE care about our users and want to help you get the most from our products.

It's Easy to Order.

Phone orders are accepted from MasterCard or VISA cardholders or for COD shipment. You can also order by mail using the coupon below. Quantity discounts are available to educational organizations and dealers. For further information contact Microware.

microware®

Specialists in system software for 68-family microprocessors since 1977.

OS-9 and BASIC09 are trademarks of Microware and Motorola.

Microware Systems Corporation
1866 N.W. 114th Street
Des Moines, Iowa 50322
Telephone 515/224-1929
Telex 910-520-2535

Please send _____ copies of the **BASIC09 Tour Guide** book at \$18.95 each. Add \$2.00 for UPS shipping in the U.S. or \$5.00 for overseas air mail per book. Iowa residents add 4% sales tax.

Name _____

Address _____

City _____

State _____ Zip _____

☐ I have enclosed a check

☐ Charge to my bank card:

MasterCard ☐ VISA ☐

Card Number _____

Expiration _____

FLEX™ USER NOTES THE 6800-6809 BOOK

By: Ronald W. Anderson

As published in 68 MICRO JOURNAL™

The publishers of 68 MICRO JOURNAL are proud to announce the publication of Ron Anderson's **FLEX USER NOTES**, in book form. This popular monthly column has been a regular feature in 68 MICRO JOURNAL SINCE 1979. It has earned the respect of thousands of 68 MICRO JOURNAL readers over the years. In fact, Ron's column has been described as the 'Bible' for 68XX users, by some of the world's leading microprocessor professionals. Now all his columns are being published, in whole, as the most needed and popular 68XX book available. Over the years Ron's column has been one of the most popular in 68 MICRO JOURNAL. And of course 68 MICRO JOURNAL is the most popular 68XX magazine published.

As a **SPECIAL BONUS** all the source listing in the book will be available on disk for the low price of: **FLEX™** format only — 5" \$12.95 — 8" \$16.95 plus \$2.50 shipping and handling, if ordered with the book. If ordered separately the price of the disks will be: 5" \$17.95 — 8" \$19.95 plus \$2.50 shipping and handling.

Listed below are a few of the **TEXT** files included in the book and on diskette.

All **TEXT** files in the book are on the disks.

LOGO.C1
MEMOVE.C1
DUMP.C1
SUBTEST.C1
TERMEN.C2
M.C2
PRINT.C3
MODEM.C2
SCIPKG.C1
U.C4
PRINT.C4
SET.C5
SETBAS1.C5

File load program to offset memory — ASM PIC
Memory move program — ASM PIC
Printer dump program — uses LOGO — ASM PIC
Simulation of 6800 code to 6809, show differences — ASM
Modem input to disk (or other port input to disk) — ASM
Output a file to modem (or another port) — ASM
Parallel (enhanced) printer driver — ASM
TTL output to CRT and modem (or other port) — ASM
Scientific math routines — PASCAL
Mini-monitor, disk resident, many useful functions — ASM
Parallel printer driver, without PFLAG — ASM
Set printer modes — ASM
Set printer modes — A-BASIC
(And many more)

Over 30 **TEXT files included in ASM (assembler) — PASCAL — PIC (position independent code) TSC BASIC-C, etc.

NOTE: .C1, .C2, etc. = Chapter 1, Chapter 2, etc.

This will be a limited run and we cannot guarantee that supplies will last long. Order now for early delivery.

Foreign Orders Add \$4.50 S/H

Softcover — Large Format

Book only: **\$7.95** + \$2.50 S/H

With disk: 5" **\$20.90** + \$2.50 S/H

With disk: 8" **\$22.90** + \$2.50 S/H

See your local \$50 dealer/bookstore or order direct from:

Computer Publishing Inc.
5900 Cassandra Smith Rd.
Hixson, TN 37343
(615) 842-4601

TELEX 558 414 PVT BTH

SOFTWARE DEVELOPERS!

YOU'VE JUST BEEN GIVEN THE BEST REASON YET
TO GET OUR 68000/UNIX[®] DEVELOPMENT SYSTEM

THE VAR/68K[®] SERIES



VK-5XW20 (List price ~~\$10,100.~~) **\$5,000.**

Includes: Terminal, 20 Mb hard disk,
512K RAM, 8 ports and REGULUS[®]

VK-5XW20T20 (List price ~~\$12,900.~~) **\$6,500.**

Includes: all of above, plus 20 Mb
tape streamer

RESELLERS!

Even more attractive specials are available to qualified resellers!

Smoke Signal has been designing, developing and manufacturing microcomputers based on the Motorola family of processors for the past six years. The VAR/68K is the most recent addition to our family of multi-user computers.

VAR/68K is a registered trademark of Smoke Signal
REGULUS is a registered trademark of Alcyon Corp.
UNIX is a registered trademark of Bell Laboratories

Due to the extremely low prices being offered, we can only accept cash or C.O.D. orders, and we must limit purchases to one per customer. This is a limited time offer.

**TO OBTAIN YOUR VAR/68K
AT THESE LOW PRICES, CONTACT:**



SMOKE SIGNAL

31336 VIA COLINAS
WESTLAKE VILLAGE, CA 91362
(818) 889-9340 / Telex 910-494-4965

Flex User Notes

Ronald W. Anderson
3540 Sturbridge Court
Ann Arbor, MI 48105

A Short Diversion

I'm going to devote a considerable part of this month's column to a discussion that might seem a bit irrelevant to the title of the column. You might consider this to be a short assessment of where we FLEX, 6809, SS-50 bus users stand with relation to the "other guys". The discussion is by no means complete. My conclusion is that there are still no newer computer systems that are appreciably better at doing the things that our systems can do, and in fact, many are not as good. However, our systems have definite limitations particularly in the area of screen graphics, an area in which more and more programs are being written, particularly in the area of education and computer aided design.

Comparison

As I reported last month, the company recently bought a Radio Shack model 1200 with a 10 MByte hard disk. The 1200 is an IBM "clone", and it was a best buy for an IBM compatible system locally. We bought it to run software from a company called Wintek. If you have been in the area of applications of 68XX microcomputers for a while, you have most likely heard of them. They have been selling 6800 hardware for a long time for industrial applications. (Coincidentally, I just today saw an ad for a 6809 processor board from them.) At any rate Wintek has been advertising a software package to run under MS-DOS for printed circuit board design. The package allows a user to prepare a PC (printed circuit) board layout on his PC (personal computer), by allowing him to place "pads" on a grid (0.05 inch) and then assisting in the routing of conductors on the board showing him both sides of the board in contrasting colors with the points where there is foil on both sides in the third color. The software finds routes for wires on the board or it will let the user route the traces (or improve on routes selected by the software). Incidentally, I think I referred to MS-DOS last month as MOOS, which of course, was Motorola's operating system for their Exorciser development system. Pardon the Freudian slip.

The system arrived this week, and it came up running. The Wintek software is Super.

It was extremely easy to learn, and is not overwhelmingly "smart". After the board layout is done, the data in the computer is used to plot artwork for the board. We have an Epson MX-100 which is usable for "prototype quality" boards, and we immediately redesigned one of our previous boards to try it out. The board took three or four days to do originally, and it took about 6 hours to do it with the computer and the Wintek software. The software, called SMARTWORK also drives a Houston pen plotter to produce superior artwork. I have already turned in a requisition for the plotter for future use. Meanwhile we will get designs done for some less critical boards and get the prototypes debugged.

Now to the point of mentioning all this (at least one of the points). After a couple days with two of us eagerly awaiting our turns at the computer to try out the software, we decided to check out the Microsoft BASIC that was purchased with the computer. The results were as expected, and a boost to our conviction that we 6809 FLEX users have good things at our disposal. I wrote a quick 25 line program to generate 1000 random integers between 0 and 999, and perform a bubble sort on them. We programmed both systems with as close to the identical program as we could. The only differences were in the use of the RANDOMIZE function in the TRS, the argument for RND(), and the call to the utility to print the clock time at the start and end of the execution of the program. The result: 6809 TSC Extended BASIC running on a 2 MHZ processor, 35.5 minutes. Model 1200 running Microsoft BASIC, 90.5 minutes! TSC Extended BASIC on the "OLD" '09 ran better than 2.5 times faster than the very new model 1200. The Microsoft BASIC is a full featured one. (The .EXE file is 66K bytes!)

As we all know, the TSC Extended BASIC is about 17K bytes. It would seem that progress must be defined as going to something more complex that is not as good (at least in terms of speed)! About here you will ask "Why did you buy the IBM clone then?" The answer to that one gets to the real heart of the question. I'll answer it by asking another question. Does anyone out there know of an interactive Printed Circuit board layout program that will generate 2X artwork on an Epson MX-100 or a pen plotter, and that runs on a 6809 system under FLEX? The Wintek folks can see the handwriting on the wall.. They have been

suppliers of 68XX products for a long while. They wrote this software to run on an IBM because everyone is buying IBM's because there's lots of software available for IBMs.. and so on in a vicious circle forever.

I'll probably use a 6809 based development system as long as 6809s are made... The processor is entirely adequate for many applications, and will not be made obsolete by 16 bit processors for a large class of applications in the foreseeable future. I'll probably not in my lifetime replace a 6809 system I have at home for word processing applications and as support for my work and consulting efforts. However, as I want to use application programs that are only available for MS-DOS or one of the -NIX variations, I'll have absolutely no choice but to switch to one of those systems for those applications. Until now, I've been content with BASIC to solve my engineering problems. Now, however, the new computers can handle something our 68XX (SS-50 version anyway) was not set up to do. Of course I am talking about GRAPHICS. To be effective, software for applications such as Printed Circuit board layout or CAD applications must deal with graphics, not only at the hard copy level (printers and plotters) but at the terminal or user interface. Try specifying a small printed circuit board layout or something as simple as a standard carriage bolt in words and you will soon understand what I am getting at.

Don't get me all wrong, our 68XX systems will do nice printer graphics, limited only by the printer's dot graphics capabilities (e.g. the turtle graphics software published in '68' Micro Journal several issues back), but there is little or no graphics capability (or multicolor display capability) at the user interface end of the system. I realize that there is at least one color graphics board available for the 68XX but there are not enough 68XX users, let alone 68XX users with that graphics board, to make it economically feasible to market anything as complex as the sort of software I've mentioned here. Incidentally the PC board software requires 192K of RAM to run.

Since writing the above, the plotter has arrived. We had no luck trying to interface it to the Tandy 1200 for a solid week. It seems that all handshake signals are not only ignored by the Tandy but cause it to hang up immediately and permanently. After

a few days of frustration, I found that I could output a "plot" file to the hard disk and then through a program in BASIC, output it to the plotter with Xon Xoff protocol via a nice feature of the BASIC. We have used that mode to plot artwork for several circuit boards. After a full week of fooling around, Tandy called us and told us that Microsoft has a problem with their operating system with regard to the serial port, and that they would get us a copy of the corrected version to try within a few days. Since we are able to use the system "as is" the pressure for a permanent solution is off. The whole problem has, however brought about some thought on my part about the situation. The Tandy system comes without any hardware documentation (schematics) and no software documentation. I suppose Tandy would say that it is a "plug it in and run" system that therefore doesn't require documentation.

Somewhere during the week, I had thoughts of a BASIC program to write 16,000 characters out to my SS-50 bus system and wait for a "send more" command. We could then use the SS-50 system to talk to the plotter, and honor all handshakes properly. Seems to be overkill to use a SS50 computer as a large buffer between the Tandy and the plotter, but I could have made it work with a half a day or so of programming. The more permanent solution from Tandy is the better way. I guess, given the choice between supposedly completely debugged software and complete documentation, I'd prefer the documentation. That way, my hands wouldn't be totally tied if things don't work "as advertised" on the first try. Certainly the availability of better documentation would have saved my employer from paying for about two man weeks of struggle in getting the software up and running properly.

Flying Lessons

I recently found a good buy on a used Atari 800 with one disk drive. My second trip to the local bookstore afterward, revealed that I could buy Atari software there (Walden Books). I am so used to buying 68XX software by mail order that the thought had never crossed my mind. At any rate, I found a good program to teach typing on the Atari and bought it for my son. At Christmas, I went back to the same store and found a Flight Simulator program. (Sorry but here we go on graphics again). If you read BYTE, you have seen the pictures

of the screen of this one or one similar. I am not a pilot, but I have flown in small airplanes with others for a long time, and I am familiar with the controls and instruments, and more particularly the radio navigation systems. The simulator comes with an aerial chart of the Chicago area down as far as Champaign Illinois. Now since I grew up in Chicago and lived in Champaign for 11 years, I was quite familiar with the route between these two points, Meigs field in Chicago, and Willard Airport in Champaign. My first efforts were to take off from Meigs, turn on the Nav. radios, and follow the Omni signals to Peotone Illinois and then to Champaign. That route takes one right down Interstate 57, which is visible during the simulated flight. The cities appear in outline form as does the Kankakee River as it is approached. All airports appear with white runways, running in the proper directions as indicated on the air chart. About all they left out is the most visible landmark in Champaign, the University of Illinois Assembly hall which closely resembles a giant flying saucer.

At an airspeed of 130 knots, it is about a 60 minute flight, and I was impressed with the passing cornfields south of Kankakee and right on down to Champaign. I managed to crash trying to land at Willard. I'm going to have to get a pilot friend to come over and give us some flying lessons. The instruments appear on the lower part of the color TV monitor, and needles move just as they do in a "live" situation. The nav. radio needle pegs realistically and overshoots as you change the heading too rapidly in an effort to center the needle. The DME equipment indicates distance to the Omni transmitter. Radios only function when within range of a station, I had the same trouble stabilizing the attitude of the simulator (oscillating between climbing and diving) as I do handling a "live" version. The program contains additional area data for the New York City to Boston area, the Los Angeles area, and the Seattle area.

Needless to say, I'm impressed, again with graphics and color displays. The market for this software in its several versions to run on Atari, Apple, and Commodore computers is so large that the price was only \$39.95! Don't hold your breath for a version to run on the SS-50 bus with a color graphics board... (After writing this today, I saw a demonstration of the program on a television show about computers, and then saw it

running at the local computer store on an Apple).

Frustrations of a Dumb Programmer

When you are a software writer, you can blame the dumb users for the problems, and when you are a user you can point at the stupid programmer for them. I am in the position of being both, so I have only myself to blame. Yes, it is time for my annual stint at printing out church contribution records for some 300 contributors. This is my third year at the job, and my software (in TSC Extended BASIC) is getting smarter each year. The biggest problem is positive identification of the contributor, and that problem can't be overcome by a smarter program. For example, some folks use a nickname or a second name in their day-to-day communications with others, but their personal checks have their "real" name on them. I may know someone as Ralph Smith, but his check might have his name as John R. Smith. It might take me a long time to make the association.

I prepare these records as though couples are filing joint returns. If half of the time the contribution is recorded as coming from Roy Jones, and the other half as coming from Susan Jones, and I happen to know that the two are married, there is no problem. If they are new, after a month or two I might begin to suspect that they are one household and that I should merge the two records.

Then there is the mobility problem. People's addresses change rapidly. To abate that problem a little, I've separated the weekly contribution entry software from the "edit the address" software.

Next problem is multiple contributions by the same person. I've made it possible to correct previous entries if I run across a second that is identifiable as being from the same contributor as the first. That is fine if there are two serial numbered envelopes in the collection with the same number, but if there is a check identifiable by name and an envelope identifiable by number, my update file building program can't correlate the two, and my program that posts the week's transactions to the main data file will post the first of the two contributions to that person's record and start a new record for the second.

As I mentioned above, each year I get smarter and make the program a little more

usable, but I still have to resort to dumping the whole data file and looking carefully for redundant records before I can actually print out the year's statements. One particularly obnoxious problem that I've had this year is that of familiarity with the data entry program. I get going fast and don't look at the prompts on the screen and type a name in a field where a number is expected. I haven't trapped the error properly and the program ceases execution with an error message and closes my data file without having written any of the data that is stored in memory, so I have to start over again. Of course some simple testing of ERL (error line number) in my error handling routine would solve that problem. That is my next project. My point is simply that handling "real world" data is not necessarily anything like the textbook examples. Making a set of programs to create, update, and maintain a database file, and to print out a report from the data in that file, is not a simple straightforward project.

K-BASIC

I have further word from Frank Hoffman at Lloyd I/O that K-BASIC is now fully capable of handling all the features of TSC Extended BASIC. The last to be added were Random Files, and PRINT USING. Frank tells me that he has removed a previous restriction of the compiler that prohibited the use of spaces in a mathematical expression. I believe K-BASIC is now fully compatible with TSC Extended BASIC, and the combination should be quite a pleasure to use. Currently, Frank has a BCD floating point math package running. It is possible to specify the number of digits of precision desired in a program. Frank also plans to have a Binary floating point package for those who want to do scientific calculations rather than financial ones, where a 9 digit binary math will offer a large speed advantage over a 15 digit BCD package. I mentioned that last month but now there is an additional possibility. The binary package may also be supplied in such a form that the user can specify the precision of the math at compile time, though that is not sure at the time of this writing.

I have received the latest version of K-BASIC since writing the above. I've compiled several TSC BASIC programs with minimal modifications, and found them to work fine. I've talked to Frank Hoffman and discussed the minor syntax differences with him. He

tells me that he will change K-BASIC to be compatible wherever there is a known difference. The latest version, as I mentioned above, has random files and "print using" implemented. K-BASIC is now so close to TSC Extended BASIC as to make conversion trivial, and I would guess that all differences will eventually be resolved so that any TSC BASIC program will compile without change, in K-BASIC.

Since writing the above on K-BASIC, the new package has arrived and I have successfully compiled a number of TSC Extended BASIC programs and reported two or three minor syntax differences to Frank Hoffman. I found that K-BASIC would accept ON ERROR GOTO <line number> but not ONERROR run together as one word. TSC will accept either and I had used both forms in various places in my programs. Changing the single word to two made the compiler accept them. I then found that EXEC,"FILENAME" worked in TSC BASIC, but only EXEC "FILENAME" (space and not comma) would work in K-BASIC. Space does not work in Extended BASIC. I had one problem with multiple statements on a line, but Frank told me that the problem has been corrected since he mailed my package. In short, there are just a few minor differences yet to be resolved to make a source listing in TSC Extended BASIC completely compatible with the compiler. At this point, the two are so close that conversion of fairly complex programs is quite easy.

Just one more note here. John Spray, author of the Whimsical language for the 6809 visited the United States early in January. John came to visit with me in Ann Arbor for a few days, and we had time to translate one of my instrument programs at work into Whimsical. I wanted to see how it would do, but this is the first time that I have had such deluxe treatment. The author of the compiler came half way around the world to give me a personal demonstration of what his compiler could do! Seriously, John and I have been communicating for a year or more regarding his compiler and my applications. We enjoyed meeting each other. Mail communications between here and New Zealand are to say the least, slow. It has taken about a month for a letter to make a "round trip". As I write this, John has returned to New Zealand (a couple of days ago). While he was here, he also visited the Williams family down in Tennessee.

OS9 USER NOTES

By: Peter Dibble
517 Goler House
Rochester, NY 14620

The Game's the Thing

I had a bunch of interesting things lined up to talk about this month: a nice sort program I've reviewed, user-written system routines, maybe something using a ring of tasks communicating with pipes. At the last minute I changed my mind.

My execution directory fills pages. I dread the times when I don't remember a program's name and have to go rummaging for it. In that entire mass of programs there is just one game (adventure -- Gimix used to include it with hard-disk systems, maybe they still do).

Microware sells a small package of games, but they make the excuse that the games are good examples of Basic09 coding technique. I guess they don't want to give anyone an opportunity to accuse them of selling recreational programs. I don't have the Microware games so I won't comment. I don't recall any other games for OS-9. Are we really such a sober-sided bunch? Anyone out there like to play?

Many of us use our computers for business. You'd think that was a good reason for the sparse supply of games, but even including the most businesslike computers I know, I can only think of two non-OS-9 machines that I didn't couldn't get games for.

Maybe the CoCo/OS-9 contingent has written a pile of games that I don't know about. The graphics and sound on a CoCo beg to have games written to use them. (Yes, I know that there are plenty of games for the CoCo without OS-9.) I went out and bought game paddles for my CoCo so I'd be ready for the games when they started arriving. No games.

I'm not going to simply complain about this. I wrote a silly little game for the CoCo. It needs lots of improvement, and it's written in Basic09 so most CoCo owners should be able to work on it. I won't get carried away with this game project; next month I'll probably get back to solid stuff again. Still, I thought up a bunch of games for OS-9 before I settled on this one. There are a bunch of old standards that wouldn't be hard to adapt. I'll come back to this again.

The game (which I think of as Weavel

or Virus) is like most shoot-em-down games. You guide your weapon around with a joy stick (it leaves a tail behind it). Circles come drifting in from the right side of the screen. If you can get the point controlled by the joy stick inside the circle before it gets to the left side or the bottom of the screen, fine. If the circle makes it to the edge you see a display that is supposed to indicate that something nasty has happened to you.

This game is pretty easy. You can make it harder by requiring a closer hit (change the constant in QHIT), or constraining the motion of the weapon. A nice constraint might be to require the weapon to be moving horizontally in order to score a hit.

Speeding the game up would help a lot. There are enough inefficiencies in the code to make that the first place to look. Recoding the program in C would probably make it fast enough so you would start looking for extra features to add. Careful assembly coding would (of course) give the best performance.

It looks hard to add sound. I wonder?

Sorry this column is so short this month. It took longer to design and write this game than I expected.

```
PROCEDURE MAIN
0000   DIM I:INTEGER
0007   DIM HISTRX(8),HISTRY(8):INTEGER
001C   DIM COLRR:INTEGER
0023   DIM BALLX,BALLY:INTEGER
002E   DIM HIT:BOOLEAN
0035   BALLX=0
003C   BALLY=0
0043   COLRL=13
004B   COLRR=13
0052   RUN INIT(COLRR)
005C   FOR I=1 TO 8
006C       HISTRX(I)=0
0077       HISTRY(I)=0
0082   NEXT I
008D   RUN FLYINGBALL(BALLX,BALLY)
009C   LOOP
009E       RUN DOMOVE(HISTRX,HISTRY,COLRR,0)
00B5   RUN QKILL(HISTRX,HISTRY,BALLX,BALLY,HIT)
00D3   IF HIT THEN
00DC       RUN EXPLODE(BALLX,BALLY)
00E8       BALLX=0
00F2       BALLY=0
00F9   ELSE
00FD       RUN FLYINGBALL(BALLX,BALLY)
010C   ENDIF
010E   ENDOOP
PROCEDURE DOMOVE
0000   DIM I:INTEGER
0007   DIM X,V:INTEGER
0012   DIM FIRE:BOOLEAN
0019   PARAM HISTX(8),HISTY(8):INTEGER
002E   PARAM COLOR:INTEGER
0035   PARAM STICK:INTEGER
003C   RUN UNDRAW(HISTX,HISTY)
0048   FOR I=1 TO 7
005B       HISTX(I)=HISTX(I+1)
006D       HISTY(I)=HISTY(I+1)
007F   NEXT I
```

```

008A    RUN GFX("JOYSTK",STICK,FIRE,X,Y)
00AC    X=4*X
00B7    Y=3*Y
00C2    HISTX(8)=X
00CD    HISTY(8)=Y
00D8    RUN DRAW(X,Y,COLOR)
00EC    IF FIRE THEN
00F5        COLOR=COLOR+1
0100        IF COLOR>15 THEN
010C            COLOR=13
0113        ENDIF
0115    ENDIF
PROCEDURE DRAW
0000    PARAM X,Y:INTEGER
0008    PARAM COLOR:INTEGER
0012    RUN GFX("LINE",X,Y,COLOR)
PROCEDURE UNDRAW
0000    PARAM HISTX(8),HISTY(8):INTEGER
0015    DIM X1,X2,Y1,Y2:INTEGER
0028    X1=HISTX(1)
0032    X2=HISTX(2)
003C    Y1=HISTY(1)
0046    Y2=HISTY(2)
0050    RUN GFX("LINE",X1,Y1,X2,Y2,12)
0073    X1=HISTX(8)
0070    Y1=HISTY(8)
0087    RUN GFX("MOVE",X1,Y1)
PROCEDURE FLYINGBALL
0000    PARAM X,Y:INTEGER
0008    RUN GFX("CIRCLE",X,Y,10,12)
0029    IF X=0 THEN
0035        IF Y<>0 THEN
0041            RUN DISPLAY
0045        ENDIF
0047        X=240
004E        Y=RND(181)
0058        IF Y<8 THEN
0064            Y=8
0068        ENDIF
0060    RUN GFX("CIRCLE",X,Y,10,15)
0088    ELSE
008F        X=X-6
009A        IF Y>10 THEN
00A6            Y=Y-2
00B1            RUN GFX("CIRCLE",X,Y,10,15)
00CF        ELSE
00D3            X=0
00DA        ENDIF
00DC    ENDIF
PROCEDURE QKILL
0000    PARAM HX(8),HY(8):INTEGER
0015    PARAM BX,BY:INTEGER
0020    PARAM HIT:BOOLEAN
0027    DIM DX,DY:REAL
0032    DIM OSQR:REAL
0039    DX=BX-HX(8)
0048    DY=BY-HY(8)
0057    OSQR=DX*DX+DY*DY
0068    HIT=OSQR<64.
PROCEDURE DISPLAY
0000    DIM I:INTEGER
0007    DIM J:INTEGER
000E    RUN GFX("COLOR",3)
001E    RUN GFX("MOVE",127,95)
0030    FOR I=10 TO 90 STEP 10
0045        RUN GFX("CIRCLE",I,3)
0058        RUN GFX("CIRCLE",I,0)
0071    NEXT I
007C    RUN GFX("COLOR",13)
PROCEDURE EXPLODE
0000    PARAM X,Y:INTEGER
0008    DIM I:INTEGER
0012    RUN GFX("CIRCLE",X,Y,10,12)
0030    FOR I=20 TO 2 STEP -2
0046        RUN GFX("CIRCLE",X,Y,I,13)
0066        RUN GFX("CIRCLE",X,Y,I,12)
0086    NEXT I
0091    X=0
0098    Y=0
PROCEDURE INIT
0000    PARAM COLOR:INTEGER
0007    RUN GFX("MODE",1,COLOR)
0018    RUN GFX("CLEAR")
0028

```

"C" User Notes

Edgar M. (Bud) Pass, Ph.D.
1454 Latta Lane
Conyers, GA 30207

INTRODUCTION

This chapter begins a tutorial on the C language, as described by Kernighan and Ritchie in their C book. The tutorial is not comprehensive or rigorous. The reader is expected to be already familiar with BASIC or similar algorithmic languages.

The only manner in which to learn any computer language, including C, is to actually write and debug a few programs. This chapter should give the reader the flavor of C. If you like it, get a C compiler for your computer and learn to use it.

GENERAL

C was developed originally at Bell Labs by Dennis Ritchie on a DEC PDP-11 under the UNIX operating system in the middle 1970's. It is so popular in its environment that UNIX itself, the UNIX C compiler, most UNIX utilities, and many UNIX application programs, are all written almost exclusively in C.

Because of its design and the method of its implementation, C has proven to be extremely portable across mainframes, minis, and, recently, micros. Unfortunately, many dialects have sprouted, and many implementations have not been very carefully done, making it somewhat difficult to transport C programs among the various implementations. However, if a subset of C is used along with programming guidelines, C programs may be generally transported far more easily than may be most BASIC programs.

DESCRIPTION

C is a "not-very-high-level" language. It generally deals with low-level concepts on the same level as macro assemblers. Thus, for example, although it deals with characters, it does not deal with strings. This type of ruthless language simplification will probably bother most programmers already familiar with APL, BASIC, or COBOL more than any other feature of the language with the possible exception of expression syntax, which is sometimes strange and arbitrary.

C provides the control structures required for structured programming, a tremendous advance over most dialects of BASIC and FORTRAN, among other non-structured languages. It requires the declaration of all variables used by the program, which prevents the problems caused by variables mis-spelled in BASIC and FORTRAN. It provides a block-structured syntax which prevents one subroutine from interfering with the variables of another subroutine, another highly-desirable feature of many structured languages. Refer often to the example at the end of this article to relate to the concepts as they are presented.

VARIABLE DECLARATIONS

As just noted, all variables must be declared before their use. Variables declared within subroutines are called "local" and those declared outside of subroutines are called "global". These terms follow the block structure of the C language and indicate the range of a variable, in terms of whether it exists for all subroutines in a program, as in BASIC, or for only the one which it is defined.

C has several types of basic variable declarations, as follows:

int	integer (usually 16 bits)
short int	short integer (usually 8 or 16 bits)
char	single character (usually 8 bits)
float	single-precision floating point (usually 32 bits)
double	double-precision floating point (usually 64 bits)
long int	long integer (usually 32 bits)

It also has a large number of variants and composite variants of these types, including "unsigned", "union", "structure", "pointer", and "function" meta-types returning these types, some of which will be covered in this chapter, and some of which will be covered in the next chapter.

An array of a certain variable type is designated by placing the constant number of elements in the array, in brackets, after the variable name, in the type statement.

Variable naming restrictions and conventions vary significantly and may cause difficulty in transporting C programs among compilers and systems. The usual mode of implementation of C compilers is to translate the C source to an intermediate assembler language, which is then assembled using either a special or a standard system assembler. Depending upon the details of the implementation, the C variables and subroutine names may be required to follow conventions and restrictions of the host system. In addition, C compilers reserve certain keywords for their own use, the exact list depending upon the implementation.

Variable and subroutine names in C should begin with a letter and may be composed of letters and numbers. Some implementations allow underline, at, and dollar as additional subsequent symbols. Some distinguish between upper and lower case letters, and some do not. Some require uniqueness in the first six characters.

Some systems require library file and function names to contain only upper-case letters and numbers, while others allow or require prefixes, suffixes, or other system-dependent information. Most full C compilers support at least the minimum

K and R standards for variable and subroutine names and they support host system standards for library files.

CONSTANTS

INTROL allows several types of and notations for constants. The user should generally avoid mixing types of variables and constants in the same expression, as the implicit conversions provided by the C compiler may not be what the user expected. For example, the following expression:

(1/2)

may evaluate to zero in many implementations, whereas the following expression:

(1.0/2.0)

will normally evaluate to 0.5 (in implementations supporting floating point, of course).

Type "int" and "float" constants are written in C, as in BASIC, using the expected notation, such as the following:

1, -1, +1, 2.5, etc.

However, the following (exponential) notation may also be used:

1E0, -1E-7, 2.5E+3, etc.

Type "double" constants are written using the same notation, assuming that a given C compiler supports them. Type "long int" constants may be written as normal integers, followed by "L" or "l". Integer constants too large to be "short int" constants will be automatically converted to "long int". Type "int" constants may also be written using octal or hexadecimal notation, by preceding a digit string with a zero for octal or by preceding a string containing digits and the letters A-F (or a-f) with "0x" for hexadecimal.

Type "char" constants may be entered as a single character enclosed in single quotes, or as a symbolic escape sequence of one of the following forms:

\n	(newline)
\t	(tab)
\b	(backspace)
\r	(carriage return)
\f	(form feed)
\l	(line feed)
\\	(backslash)
\'	(single quote)
\0	(null)
\nnn	(octal nnn)
\xnn	(hex nn)

Type "char" constants may be used wherever "int" constants may be used. However, various implementations vary concerning the sign-extension or truncation employed in the conversion of type "char" to type "int".

A string constant, as in BASIC, is a series of zero or more characters (or symbolic escape sequences) surrounded by double quotes.

FUNCTIONS

A C program is a series of one or more function definitions. By convention, the first (or only) function executed is named "main". K and R provides the following example of a C program, which happens also to be exactly one function:

```
main ()
{
    printf ("hello, world\n");
}
```

This trivial C program is intended to print the following:

hello, world

and illustrates, by example, several points about functions. The body of "main" is enclosed in open and close braces, as required for all functions. The library function "printf" is called with one string constant argument. The function call statement is followed by a semicolon.

The function "main" (as shown here) has no parameters, but, if it did, they would be listed between the parentheses, separated by commas, and declared as local variables, as in the following function definition example:

```
add1(n)
int n;
{
    n = n + 1;
    return (n);
}
```

which also illustrates several items concerning functions. The primary point is that functions can return values; if a function does not have an explicit type, it is defaulted to type "int".

The only manner in which a function can pass a value directly to a caller is thru the parameterized option of the "return" statement. However, a function is not required to have a type, as shown by the call to "printf" in the original example. Since the variable "n" is local to function "add1", no value may be directly passed back thru a parameter. Technically, all C parameters are passed by value, not by address, making it more difficult to pass results back to the caller, but solving many "side-effect" problems.

Results may be returned to the caller thru globals, arrays, or pointers, among other methods. However, this method of parameter passing avoids the FORTRAN horror of accidentally modifying constants, as in the case of passing a constant, not a variable, to the "add1" function.

The expandability of C is partially due to the ease of definition of C functions in a private library; "printf" is one such library function. The standard C library will be discussed in a later chapter.

EXPRESSIONS

Although expressions in C may look a little strange at first to someone familiar with BASIC, there are strong parallels between the languages in terms of expression formation.

The arithmetic operators are as follows:

+	addition
-	subtraction
*	multiplication
/	division
%	modulus
-	unary negation
++	unary increment
--	unary decrement
,	sequential evaluation

The logical operators are as follows:

&&	and
	or
!	unary not

The relational operators are as follows:

>	greater
>=	not less
<	less
<=	not greater
==	equal
!=	not equal

Note that the "equal" operator is "==", not "=".

The assignment operator "=" is allowed inside expressions, and has the same interpretation as outside expressions, of changing the value of the variable on its left side to that of the expression on its right.

The bitwise logical operators are as follows:

&	and
	or
^	exclusive or
<<	left shift
>>	right shift
~	unary one's complement

The conditional expression is designated by the pair of operators "?" and ":" when used in the following context:

e1 ? e2 : e3

which may be interpreted as follows:

if "e1" is true
the value of the expression is "e2"
else
the value of the expression is "e3".

The value of a "true" logical expression is usually non-zero (one in some implementations, minus one in some, other values in some), and the value of a "false" logical expression is usually zero (minus one in some implementations). Thus logical expressions may be used in arithmetic expressions, with care.

Parentheses may be used, as in BASIC, to force grouping. Brackets are used to indicate array subscripts, rather than the

double use of parentheses in BASIC and FORTRAN.

The decrement and increment operators of C look strange initially to BASIC programmers. The increment operator "++" adds one to a variable and the decrement operator "--" subtracts one from a variable. If the operator appears before the variable, the variable is incremented before using its value, and if the operator appears after the variable, the variable is incremented after using its value.

For example, the following statement:

```
if (n < 10) ++n;
```

would cause "n" to be incremented by one if its value were less than 10, as would the following statement:

```
if (n < 10) n++;
```

since the effect on "n" is the same in both cases. But, in the following statement:

```
if (n < 10) a[n++] = n;
```

the value of "n" is incremented between uses, storing the value of "n" in a different element of "a[]".

There is an option of the assignment operator "=" analogous to the increment and decrement operators. If the "=" is preceded by one of the following binary operators (call it "b"):

```
+, -, *, /, %, <<, >>, &, ^, |
```

the compound assignment resulting from "b=" in the following expression:

```
u b= expr
```

is equivalent to the following expression:

```
u = (u) b (expr)
```

so that the following (commonly used) expression:

```
w += 2
```

is equivalent to the following expression:

```
w = w + 2
```

in all contexts.

The "comma" operator is used in two contexts. The most familiar one is that of separating parameters of a function call or definition or of separating items being declared. The other use is that of causing sequential evaluation of arithmetic expressions; the value and type of the last expression is used as the value and type of the entire group. It has the following format:

```
(expl,...,expn)
```

Although the parentheses are not a part of the comma operator, they should always be coded to signify that the commas are used in this context and not as separators.

The fact that the assignment operator may be used in an expression leads to programmers often confusing it with the equal relational operator. Former BASIC programmers must be especially careful not to write statements such as the following:

```
if (c = 2) n = 3;
```

which seems natural but was probably intended to be the following:

```
if (c == 2) n = 3;
```

One hint is to write constants first, whenever possible, in relational expressions; then the following expression would generate a syntax error:

```
if (2 = c) n = 3;
```

rather than compiling quietly, but perhaps incorrectly.

OPERATOR PRIORITIES AND ASSOCIATIVITIES

One of the most troublesome "features" of the C language is its somewhat strange set of rules of operator hierarchy (priority) and associativity (left to right or right to left). The following table provides this information, with the rows arranged in order of decreasing priority:

OPERATOR	ASSOCIATIVITY
{ } [] -> .	left-right
! ++ -- -()	right-left
(type) * & sizeof	
* / %	left-right
+ -	left-right
<< >>	left-right
< <= > >=	left-right
== !=	left-right
&	left-right
	left-right
^	left-right
&&	left-right
!!	left-right
?:	right-left
= += -= /= %= (etc.)	right-left
,	left-right

Some of the operators in this table are not in the "natural" order of most other languages. To avoid undesired operator grouping, parenthesize heavily. Among expressions involving operators of equal priority, K and R does not specify order of evaluation; thus avoid expressions such as the following:

```
(a[n++] + n++)
```

which has different interpretations, depending upon whether C evaluates the expression left-to-right or right-to-left.

STATEMENTS

The C language has only a limited number of statements, far fewer than BASIC or FORTRAN, since C relies on function libraries to implement all I/O and other system interfaces. Several functions have

already been used without explanation to avoid cluttering this explanation with side-notes. C uses semicolons to separate statements, whether on the same or on different lines.

The simplest (and sometimes most treacherous) C statement is the comment statement. It is introduced with `/*` and terminated with `*/`. Comments may officially not be nested (according to K and R), although some versions of C (including BDS and INTROL) allow nested comments. Since C is normally insensitive to end-of-line in many cases, a missing or miscoded `*/` can cause large chunks of a C program to be ignored incorrectly. The best advice on comments is probably to include the termination on the same line as the introduction.

In several cases, expressions may appear as statements. Several instances of this have already been shown. The primary situations in which expressions are used as statements include the following:

- assignment
- function calls
- increment
- decrement
- compound assignment

The "if" statement is somewhat similar to the BASIC "if" statement. Its syntax is as follows:

```
if (expression) statement
if (expression) statement else statement
```

Note that, unlike BASIC, parentheses are required surrounding the control expression. The interpretation of the "if" statement is identical in C and in BASIC. However, the use of Boolean expressions in BASIC may not be compatible with the definitions of TRUE and FALSE in a specific C implementation.

The "while" statement establishes a structured loop in which the control condition is tested before the first (and any subsequent) executions of the body of the loop. Its syntax is as follows:

```
while (expression) statement
```

and it may be represented symbolically by a BASIC sequence similar to the following:

```
100 if (expression) goto 200
    goto 300
200 statement
    goto 100
300 ...
```

The "do" statement establishes a structured loop in which the control condition is tested after the first (and any subsequent) executions of the body of the loop. Its syntax is as follows:

```
do statement while (expression)
```

and it may be represented symbolically by a BASIC sequence similar to the following:

```
100 statement
    if (expression) goto 100
```

The "for" statement establishes a loop based upon three control expressions. Its syntax is as follows:

```
for (expression1; expression2; expression3)
    statement
```

and it may be represented symbolically by the BASIC sequence similar to the following:

```
expression1
100 if (expression2) goto 200
    goto 300
200 statement1
    expression3
    goto 100
300 ...
```

Note that the C version of the "for" statement is far more sophisticated than is the BASIC version. Its interpretation is also different, in that it checks the condition the first time, as opposed to BASIC, which does not check the condition the first time.

The "switch" statement provides a multi-way decision scheme similar to an iterated

```
if (...) ... else ..
```

statement. Its syntax is as follows:

```
switch (expression)
{
    case C1: statement; ....
    case C2: statement; ....
    :
    :
    default: statement; ....
}
```

where the "C1", "C2", etc. represent constants with the same type as the expression and the "default" clause is optional. The expression is evaluated and matched against "C1", "C2", etc. If a match is found, that sequence of statements is executed. If not, the "default" sequence is executed. If a match is found, a "break", "continue", or "goto" is normally required to prevent falling thru to the next case.

The "break" statement causes an immediate exit from the innermost "do", "for", "switch", or "while" statement in which it appears. It functions as if it were a "goto" statement referencing an imaginary label just beyond the end of the statement. Its syntax is as follows:

```
break
```

The "continue" statement causes the next iteration of the innermost "do", "for", or "while" statement to be performed or the loop to be terminated, as the control expression dictates. Its syntax is as follows:

```
continue
```

The "return" statement has already been discussed. It is used to return control to the caller of a function. Its syntax is as follows:

```

return
or
return (expression)

```

where "expression" provides the value returned by the function, if any. If a value is expected, but none is provided, the resulting value is unpredictable. If no value is expected but one is provided, it is discarded.

The "goto" statement is used to explicitly transfer program control to a label. The syntax of a "goto" is as follows:

```
goto label
```

and the syntax of a label is as follows:

```
label: statement
```

where "label" must follow the rules for C variables and must be defined within the same subroutine or block in which it is used.

The "null" statement is allowed and is useful in many cases in which a statement is required, but is not needed, such as in "for" and "while" statements.

The C language allows a compound statement to appear wherever a statement may appear. This is indicated by the use of open and close braces ("{" and "}") delimiting a sequence of zero or more declarations and zero or more statements. The syntax for the compound statement is as follows:

```

{
    declaration1;
    :
    :
    declarationn;
    statement1;
    :
    :
    statementn;
}

```

PREPROCESSOR

Most C compilers provide a preprocessing capability which extends the compiler's usefulness in a manner similar to that provided by macro assemblers. Lines beginning with "#" are preprocessor definition lines.

Although there are many preprocessor definition commands, only the two most common ones are described below.

The most commonly-used preprocessor command is "#define". Its syntax is as follows:

```

#define identifier string
#define identifier (idl,...,idn) string

```

In the first case, subsequent occurrences of "identifier", not in comments or string constants, are replaced by "string". In the second case, subsequent occurrences of "identifier" followed by the specified parameter list are replaced by "string", modified by parametric replacement of the "idl" thru "idn" in "string". This capability is used primarily for cases such as the following:

```

#define tabsize 2000
:
:
int tab[tabsize];

```

since the size of an array is required to be a constant.

Another commonly-used preprocessor command is "include". Its syntax is as follows:

```
#include "filename"
```

This causes the preprocessor to search for "filename" in an operating-system dependent manner, and replace the "include" command with the entire contents of the file, which is, of course, assumed to contain C functions, declarations, statements, etc. Some implementations allow or require alternate syntax of the "#include" statement, primarily concerning the inclusion, deletion, or alternation of the delimiters surrounding the filename.

SUMMARY

This chapter began a tutorial on a subset of the C programming language. Subsequent chapters present some of the essential language and library elements not described in this chapter.

The reader should be able to understand the example below. If you have access to a C compiler, key in the example and run it. The only point in the example not covered in this chapter is that "l3d" in a "printf" string constant causes a number to be printed in decimal notation. This will be covered in a later chapter.

```

/* Eratosthenes Sieve Prime Number Program in C */
/* from Byte, September 1981, Pg. 186 */

```

```

#define TRUE      1
#define FALSE     0
#define SIZE      8190
#define SIZEPL    8191

char flags[SIZEPL];

main()
{
    int i,prime,k,count,iter;
    printf("10 iterations\n");
    for(iter=1; iter<=10; iter++)
    {
        count=0;
        for(i=0; i<=SIZE; i++) flags[i]=TRUE;
        for(i=0; i<=SIZE; i++)
        {
            if(flags[i])
            {
                prime=i+1+3;
                k=i+prime;
                while(k<=SIZE)
                {
                    flags[k]=FALSE;
                    k+=prime;
                }
                count=count+1;
            }
        }
    }
    printf("%d primes\n",count);
}

```

68000 USER NOTES

Phillip Lucido
2320 Saratoga Drive
Sharpville, PA 16150

More on the 68020

This, hopefully, should finish up my columns on the 68020. This month the focus is more on the internal and hardware aspects, after last month's software emphasis on new instructions and addressing modes. I'll mostly just be going through the 68020 user's manual, picking out the high points.

Registers

The 68020 adds a few new registers to the large complement of the 68000. Address register A7, which functions as the stack pointer, is actually two separate registers in the 68000, one each for the user and system states. The 68020 has three A7 registers, with two now devoted to the system state. These are the master and interrupt stack pointers. The actual register to use is determined by a new M bit in the status register. The reason for two system stack pointers is to simplify processing in multi-tasking environments. The interrupt stack pointer (ISP) is enabled by default after reset or after an interrupt. The master stack pointer (MSP), if enabled by setting the M bit, will be active after a non-interrupt exception, such as a TRAP call from the user state. This enables an operating system to set up a separate system stack for each executing process while multi-tasking, speeding the processing of system calls, but use only one stack for interrupt processing.

In addition to the new M bit in the status register, there are now two trace bits, T0 and T1, instead of the single T trace bit in the 68000 SR. The additional trace bit is used to only trace instructions which change the flow of a program, like JMP, JSR, TRAP, instead of tracing after each instruction.

There are three registers which were actually first introduced on the 68010. These are the vector base register (VBR), and two alternate function code registers (SFC and DFC). The VBR is used to supply an origin for the 1K table of exception vectors, which is found at address 0 by default in a 68000. This allows separate exception tables to be defined for each process in a multi-tasking system. The

alternate function codes are used by the MOVES instruction, which allows access to an address space (such as user data space) not normally accessible to the system state.

Instruction Cache

The 68020 has a cache of on-chip memory which allows tight loops and commonly used instruction sequences to execute without reading the external memory. The cache is 256 bytes long, organized as 64 long words. The cache is only used during instruction fetches, so data references always go to the external memory. Since actual programs spend most of their time executing a small number of loops, this cache can significantly speed execution, while freeing the external bus for use by other processors or DMA.

Two new registers control the cache. The cache control register (CACR) has four bits used to modify the cache's behavior. The E bit is used to enable cache operation under software control. The F bit freezes the cache, so it's current contents are not updated. The C bit is used to clear the current cache contents. Finally, the CE bit, along with the new cache address register (CAAR), is used to clear a single entry in the cache. In addition to these software controls, there is a single input signal, CDIS/, used to disable the cache under hardware control.

Virtual operation

In common with the 68010 and 68012, the 68020 is capable of what is called 'virtual memory'. With virtual memory, a user program may be written as if it had a huge amount of memory available to it (up to the 4 gigabyte addressing range of the 68020), while in reality a much smaller amount of memory is actually present in the computer.

In virtual memory, the system program allocates some physical memory to a user program, while keeping track of all memory which the user thinks it has access to. The user program issues memory references using addresses of memory which it believes is available. If an address corresponds to physical memory currently allocated to the user program, then the user address is translated to the proper physical address and the operation is completed. This translation is performed automatically by a memory management unit (MMU). If the user address does not so correspond, though,

then the MMU issues a 'page fault', which instructs the system program to load the referenced memory from secondary storage, such as a hard disk. By keeping most of a user program's memory on disk, rather than in immediately accessible memory, a user program can be made to believe it has nearly unlimited resources, while keeping the actual system memory down to some more reasonable figure.

In the 68020, this virtual memory is supported by being able to suspend an instruction in the middle of its processing. If the MMU signals the 68020 that a page fault has occurred, the current processor state is completely stacked. The information stacked includes every internal register needed by the 68020 to determine just where processing was halted. The system program looks at this information to determine the memory to be loaded from the disk, then executes an RTE (return from exception) instruction. The RTE reloads the processor state, and the instruction is continued from where it left off.

This ability to continue an instruction after it has been halted by non-existent memory is also useful for emulating virtual I/O devices. When a user program references a particular address, which is assigned to an I/O device not actually in the computer, the system program can detect this fact and simulate the device. Thus, the user program never knows that the device is not present.

Addressing and the Bus

The 68020 has loosened the constraints governing access to memory through the bus. In the 68000, all references to words or long words in memory were required to be aligned to an even byte address. In the 68020, while the restriction still exists for instructions, it is no longer in effect for data references. Thus, for instance, a long word MOVE to memory may now be executed to any byte address.

Writing or reading long word data with a misaligned address, with a full 32 bit data bus, requires splitting the write or read into multiple cycles. Thus if the long word is being stored at an address of \$xxxxxxx1, the high three bytes are first written on the lower 24 bits of the data bus, with the upper 8 bits to be ignored, followed by writing the low byte on the upper 8 bits of

the data bus, with the lower 24 bits to be ignored. To perform this operation, the 68020 supplies two output lines, SI20 and SI21, which describe the size of the data for the current operation. In conjunction with the actual address on the bus, memory must decide which data bits are actually active, and which are to be ignored.

The 68020 also allows dynamic bus sizing. This means that the memory connected to the 68020 may be organized as 8-, 16-, or 32-bits wide. With each transfer, the addressed memory tells the 68020 how many bits it can handle at a time, using two input lines, DSACK0/ and DSACK1/. The 68020 will automatically split the reference so that data is transferred using the proper number of bits.

By combining misaligned references with dynamic bus sizing, it is obvious that the problem of deciding just what is present where on the data bus can get very complicated. Memory must be quite a bit smarter than at present, but in exchange the maximum flexibility in addressing is possible.

Instruction Timing

The 68020 is equipped with a three stage pipeline, which enables it to prefetch instructions well before they are actually executed. In addition, the internal processing is divided up so that at a single instant, one instruction might be writing to memory, the following instruction is performing arithmetic with the data registers, and the instruction after that is being decoded to prepare for execution.

This concurrency of operation, overlapping stages in the execution of instructions, means that instruction timing is no longer an exact thing. Add to this the presence of the instruction cache, so that an instruction may or may not require an external memory read, as well as misaligned accesses and dynamic bus sizing. The most that can be said about instruction timing is now to give a best and worst case timing, with the average falling somewhere in between, depending on the context of the instruction. The difference between best and worst case is often seven cycles or more. With caching and overlap, some instructions may even appear to execute in zero cycles.

And So On, Forever More

If by now you've gotten the impression that the 68020 is one powerful, but complicated, microprocessor, congratulations! You're right on the mark. I'll cut it short here, even though I've only scratched the surface. There is still the coprocessor interface, the module call support, address space access levels, and various new exception types. Enough is enough, though. If you want to know more, see the book I've been cribbing from for the past few columns, the **68020 32-Bit Microprocessor User's Manual**, Motorola part number MC68020UM(ADI).

On the Home Front

Currently, I am a system programmer for a company involved in application programming for various niches like financial computing. Unfortunately, I've decided that my idea of heaven on earth involves working at home for myself, doing freelance programming, possibly as a consultant but preferably writing programs for the general market. To that end, I've decided to go off on my own, about six months from now.

What does this have to do with this column? Well, I'm not sure what kind of living can be made from writing software just for the 6809 and 68000 using the CoCo or OS-9, so I've plunked down my money and bought a Macintosh.

Am I abandoning the S-50 and 6809? Not at all. In fact, I am right now working on a few projects which will, with a little luck, do fairly well. Still, there are at least 300,000 Macs out there. It is just a little hard to ignore that large a user base if I have any hope of supporting myself with freelance software.

In addition to the sales aspect, the Macintosh is simply a fun machine. It really is radically different from the sort of computer I'm accustomed to using. For one thing, I've never done much with graphics, which is at the heart of the Mac. The user interface, with the mouse and the desktop icons, makes use of the machine a breeze. The Macintosh is not all perfection, by any means. I am at heart a hacker, while the Mac is a user's machine. I don't foresee my Helix gathering dust in some corner, since it really helps fulfill a quite different type of 'computer experience.' In the same way, I can easily see, say, the 68000 with OS-9/68K doing quite nicely, occupying a different part of the market. I don't think that the Mac is the way to go for real-time

data logging, for instance. Also, the Macintosh can be unbearably slow at times, what with its small capacity (400K) disk drives. I'm not sure that a hard disk drive would completely solve my dislikes, either. Anyway, these are just some initial thoughts. I've had my Mac for all of two days now, which hardly makes me an expert. All the software I have so far consists of MacPaint and MacWrite, some interesting public domain stuff, and the Sargon III chess program (which either plays a terrific game of chess, or I'm much rustier than I thought - probably both). I've got a C compiler, which includes an assembler, coming in soon, so I should be able to get some real work going.

Will this column cover only the Macintosh from now on? No, since I really do plan on using my Helix. Instead, expect to see a mixture, with OS-9/68K, the Mac, and 68000 material in general. When I do cover the Mac, I'd prefer to get technical, since many of you are also hard core hackers, who know that any machine, no matter how user friendly, can be ripped to pieces for fun and profit (figuratively of course - I've spent too much to trash my Macintosh just yet).

Whimsical Revisited

A year or so ago, I reviewed a new compiler called Whimsical. For those of you who don't have that review, I will repeat the description of this language. Whimsical is not a standard language, but it very closely resembles Pascal in many ways. The reason it is different from the standard languages, is that it was "tailored" to fit the structure of the 6809 processor (and I might add, to some extent, stand-alone system applications as in controls and instrumentation). To quote the instruction manual, "The three lexical levels of Whimsical are chosen to be efficiently supported by the 6809's complement of index registers. Also 8 bit arithmetic and fast unsigned multiply are supported directly."

When I reviewed Whimsical previously, it supported only integer arithmetic (in several sizes and varieties, as it does presently). I don't have that review in front of me as I write this, but at the end I said something like "Now if you only would add floating point capability...." Well, the author of Whimsical, John Spray, of Auckland New Zealand, has done just that. In addition, he has made several changes that have added to the utility of Whimsical. The floating point arithmetic is the more or less standard 24 bit signed mantissa and 8 bit exponent form, giving almost 7 digits precision. I find this level of precision adequate for nearly all machine control and measurement applications.

The best new feature, however, is the ability to compile separate code modules which may then be included in programs. Each module requires a simple header that declares the GLOBAL variables and lists the include procedures that are to be "Publicly" accessible, using the keyword "PUBLIC", and the LOCAL variables using the keyword "PRIVATE". If the module requires access to variables declared elsewhere in a program, those variables must be listed under the keyword "EXTERNAL". This may

occur if you pass parameters to the procedures in the module via GLOBAL variables.

Another recent modification was to include all the normal runtime subroutines with the compiler file so that they all load into memory when the compiler is loaded. This avoids disk searches for subroutine files during the compile. Those that are needed are simply copied from memory to the output file. Because of these two new features, I found that the compilation time is greatly reduced. The source code to several of the runtime subroutines is provided for information, and if you need a custom version of one or more of these, there is an option that causes the compiler to scan the working disk for your customized subroutines, using them in preference to the built in ones.

I've included a listing of a program that contains the procedures necessary to calculate the sine of an angle, and a "main" program that is essentially a test program to allow me to "exercise" the sine procedure to test for errors. The program contains the instruction to include the module "FIXEDIO" which contains the code necessary to allow formatted output of floating point numbers in a manner similar to Pascal. That is, the procedure WRITER (for WRITE Real) is passed the variable name and specification for field width and digits after the decimal point, just as in Pascal. Since the code in FIXEDIO was supplied by Whimsical Developments as part of a package of "library" modules, and since I have only modified it slightly, I won't reproduce it here. You might note the declaration of an array of REAL constants that are used in the calculation. To my knowledge, the only other programming language that allows this is "C".

Compilation of the SCITEST program took a total of 18 seconds including the time to load the compiler, get the FIXEDIO module, compile the program and write the object file to disk! That time is for a 2 Mhz system. Of that time, about 7 seconds is used just to load the compiler, which with all the subroutines, is now 129 sectors long. I also tested the compile time on a 1 Hz system, and found it to be 28 seconds, still very reasonable as compile times go.

The compiler supports three sizes of signed integer variables: SMALLINT (8 bits), INTEGER (16 bits) and LARGEINT (32 bits), as well as the REAL data type. There are two unsigned integer types, BYTE and DBYTE. These two types always use hexadecimal format for input, output, and literal constants. DBYTES are of course very useful for Address information. There are also types for CHAR and BOOLEAN. The data typing is strict, and there are "intrinsic" functions provided for each of the type conversions. There are no automatic type conversions. By this, I mean that, for example, if a constant is declared BYTE OR = \$0D; you cannot WRITE CR to the terminal. You must "convert" it to a CHAR type by using the intrinsic function CHR as in WRITE CHR(CR). The function CHR does nothing other than to signal the compiler that you haven't unintentionally used a constant of the wrong type. Whimsical is very "Pascal like" in this respect. This example is not a very good one since you could simply declare the constant as CR = CHR(\$0D), and you would have a constant of type CHAR. However, it illustrates the point.

Full arithmetic overflow checking may be provided at runtime by including the optional runtime checking when the program is compiled. Optionally, overflow may be handled by "saturation". That is, for example, SMALLINT has the range of -128 to +127. If the value +130 is put into a SMALLINT, it may be made to "saturate" and assume the value of +127. Normally, overflow would cause it to assume the value -126 for such an overflow condition. Obviously, +127 is closer to the desired value than -126. An error may be flagged on overflow regardless of whether the saturation option is in effect. Singly dimensioned arrays, sometimes called vectors are supported, and full support for sequential files is provided.

This is excellent software. It is efficient and well debugged. You will have little trouble learning it if you have a working knowledge of Pascal and just a little more difficulty if you know "C". There are a few peculiar syntax differences between Whimsical and Pascal that are quite obvious, perhaps purposely so that you can distinguish this code from Pascal at a glance. The most obvious ones are the use of a percent sign (%) in the leftmost column as a delimiter for a comment line, and that a Procedure declaration is terminated with an equal sign (=) rather than a semicolon (;) as in Pascal. The usage seems rather natural. It is as though you are declaring the Procedure to be equal to the statements that follow the header.

PROCEDURE GETCHAR=

There is one abbreviation allowed. If you like, you may substitute "curly braces" for BEGIN and END respectively.

I've tried here to cover all the interesting and unusual points about Whimsical with an emphasis on the new features that have been added since the previous review. I'm sure I've overlooked some and valuable feature somewhere, and that you readers and/or John Spray will let me know after reading this.

Review by: Ron Anderson

Whimsical is a product of:

Whimsical Developments
P.O.Box 47-121,
Auckland, New Zealand
It is available from Southeast Media.

(see Advertising this Issue)
FREE CALL

1-800-3 8-6800
Price: \$195.00

- - -

```

$ MODULE FOR SINE FUNCTION
MODULE SINMOD=
BEGIN
  PUBLIC
    REAL PI02 = 3.5707963;
    REAL PROCEDURE SIN(REAL OPER);

  PRIVATE

REAL PROCEDURE POLY(REAL OPER; REAL ARRAY TABLE; SMALLINT COUNT);

BEGIN
  POLY := TABLE[COUNT];
  DO BEGIN
    COUNT := COUNT-1;
    POLY := POLY+OPER*TABLE[COUNT];
  END UNTIL COUNT=0;
END;

REAL PROCEDURE SIN(REAL OPER);
BEGIN
  BOOLEAN NEGATIVE;
  SMALLINT IQUAD;
  REAL QUADRANT;
  REAL ARRAY SIN_COEFF=( 1.0, -0.1666666, 8.333332E-03,
                        -1.9852E-04, 2.8255E-06, -3.7E-08);

  IF OPER<0.0 THEN
  BEGIN
    QUADRANT := FLOAT(INT(OPER/PI02));
    OPER := OPER-QUADRANT * PI02;
    IQUAD := TRIM(TRIM(FIX(QUADRANT)));
    IQUAD := IQUAD MOD 4;
    NEGATIVE := (IQUAD=2 OR IQUAD=3);
    IF IQUAD=1 OR IQUAD=3 THEN OPER := -PI02 - OPER;
    SIN := OPER * POLY(OPER+OPER*SIN_COEFF,5);
    IF NEGATIVE THEN SIN := -SIN;
  END;
END;
END.

$ ATTEMPT AT SCI PACKAGE IN WHIMSICAL
*ORIGIN = ($ODD0);
*STACK = ($OFF);

BEGIN
  REAL NUMBER, EXPONENT;

  MODULE SINMOD = CODE FROM "SINMOD.MOD.1";
  MODULE FIXEDIO = CODE FROM "FIXEDIO.MOD.1";

  $ MAIN PROGRAM HERE
  DO
  BEGIN
    WRITE "INPUT ANGLE FOR SINE FUNCTION (RADIANS)";
    READ NUMBER;
    CRLF;
    WRITE "SINE IS ";
    WRITER(SIN(NUMBER),12,6);
    CRLF;
  END
  UNTIL NUMBER = 0.0;
END.

```

ADA^R AND THE 68000

By:
THEODORE F. ELBERT
THE UNIVERSITY OF WEST FLORIDA
PENSACOLA, FLORIDA 32514

PART I - WHAT IS ADA?

In the ten years since the introduction of the 8080 as the first really useful microprocessor, hardware technology has advanced at a most astounding rate. While this rapid advancement in hardware capability originally held promise of making software development easier, just the opposite effect was noted. As the computational power of computers grew, the applications grew in proportion -- with the inevitable result that software development became the critical shortcoming of the industry. For a number of years, software technology -- in the form of the emerging field of software engineering -- grew at a pace much slower than that exhibited by its hardware counterpart. Part of this effect was due to the characteristics of existing programming languages, in that their design did not directly support many of the software engineering practices being developed as solutions to the software problems. Within the Department of Defense this trend was becoming increasingly evident, in the form of software that was very expensive, and yet was unreliable. Spending more money on software was not the answer, since reliability of software seemed nearly independent of the cost.

In addition to the computers getting more powerful, they were also getting less expensive. When compared with the cost of software -- which, being labor intensive, steadily increases in cost -- the decreasing cost of hardware produced an overall system cost with a constantly increasing software component. When life-cycle costs were considered, the effect was even more pronounced, since poor reliability was making software maintenance a major factor. In the mid 1970's, the Department of Defense embarked on the development of a programming language that would support and encourage the use of modern software engineering concepts such as abstraction, information hiding, and modularity. The result is the Ada programming language, established as an ANSI standard in February, 1983. Since that time, several Ada compilers -- some of them with associated work stations -- have been developed. Many

of these compilers target the Motorola 68000 family of microprocessors, and many of the work stations are 68000 based. In this series of articles, the salient features of the Ada language will be presented, and the role played by the 68000 family of microprocessors in the developing Ada story will be investigated.

The United States Department of Defense is the worlds largest user of computers. The mounting problem of software development was of critical concern to the Department for two reasons. First, of course, is the fact that software reliability directly affects weapon system reliability -- a critical concern to defense planners. The second reason involved the ever present budgetary consideration -- software costs were becoming the principle factor in computer system costs, and computer systems costs were becoming an appreciable part of the overall cost of any modern weapon system. Coupled with the fact that -- no matter what the cost -- software was not performing reliably, these considerations prompted a move on the part of the Department of Defense to actively counter the alarming trends in software development. To properly present the problem as it existed at the time, it is necessary to digress slightly and consider what is meant by the term embedded computer system.

Embedded computer systems are usually defined to be those computer systems that constitute part of a larger system whose primary function is other than computational. In the case of the Department of Defense, the application is clearly to weapon systems and other military uses, but the general concept of an embedded computer system applies equally well to process control, to communications systems, and to many other non-military uses of computers. Within the Department of Defense, and in other areas as well, embedded computer systems generally have the following characteristics:

- Programs tend to be large, on the order of tens of thousands or even hundreds of thousands of lines of code.
- They are in service for extended periods, perhaps up to twenty years.
- They must be fault-tolerant, recovering from faults or gracefully degrading to a lower level of performance. This includes software response to exceptions in the

processing.

- They undergo continuous field changes, including software updates due to design improvements or changing operational requirements.
- They must have high reliability, both in hardware and in software.
- There are usually physical constraints in terms of hardware size or processing speed, implying a requirement for efficiency in the software.
- There is usually a requirement for concurrent processing.
- Input-output requirements are usually specialized.
- There is usually a requirement for real-time processing, in that the system must respond to physical stimuli in real time.
- There is normally a host-target relationship, with software development taking place on a large host computer equipped with ample peripherals and with software development tools, while the actual application is to the embedded target computer containing only those features necessary for mission functionality.

It was the embedded computer system that was targeted by the Department of Defense in its effort to improve the quality and reduce the life-cycle cost of software. The reason for this orientation was the observation that software life-cycle costs were becoming a major contributor to overall systems costs, and that the cost of software maintenance was often exceeding that of initial software development.

Studies conducted in the mid 1970's indicated that software developed for Department of Defense embedded computer systems was often non-responsive to user needs, tended to be unreliable in spite of high cost, was inflexible and difficult to maintain, and was not portable or reusable. In particular, programs written in assembly languages -- by their very nature -- were not portable and were difficult to maintain. The first step towards effecting some control over the software problem was the issuance of two directives:

- DOD Directive 5000.29, April, 1976 - required the use of a DOD approved high order language in defense systems. This directive essentially prohibited the use of assembly languages.
- DOD Directive 5000.31, November, 1976 - reduced the list of DOD approved high level languages to seven

- FORTRAN - SDL/1
- COBOL - JOVIAL J3
- TACPOL - JOVIAL J73
- CMS-2:

These directives put an end to the use of assembly language -- although waivers could be obtained for justified reasons -- and, perhaps more importantly, curtailed the proliferation of high order languages.

The second step in the process was to determine just what characteristics a high order language for embedded system programming should exhibit. While it was recognized that a programming language is neither the cause of nor the solution to software development problems, it was also realized that a language can promote the application of the sound software engineering practices that were seen as the best approach to the solution of the software problem. These desired characteristics were compared to existing compiler design technology -- in order to ensure feasibility -- and the results were incorporated into a set of language requirements. The requirements document went through three iterations, aptly named

STRAWMAN	April, 1975
WOODENMAN	August, 1975
TINMAN	June, 1976.:

The TINMAN requirements were compared to existing high order languages, with the following conclusions:

- None of the existing languages met the requirements.

- A single language that met the requirements could be developed.

These conclusions formed the conception of the Ada Language. Further refinements of the requirements documents,

IRONMAN	January, 1977
STEELMAN	June, 1978

were issued, the latter becoming the design goal of the language development effort.

The actual development process began with the request for proposal, issued in January, 1977. The bidders were instructed to begin with a base language, selected from a group of three:

- Pascal
- ALGOL 68
- PL/1.

The language development proceeded as a two-phase competitive effort, with four contractors selected for the first phase and the field narrowed to two for the second phase. The final result was based on the STEELMAN requirements, was based on Pascal as the base language, and was the

product of the French firm Honeywell/Bull.

The final language design represented the culmination of six years of effort on the part of the Department of Defense and its contractors. There were five refinements of the technical requirements; there was scientific cooperation among military, industrial, and academic communities unprecedented in any previous language design effort; there was an integration of international interests, with representatives of fifteen countries involved; and the design process was open to continuous public scientific scrutiny. The Ada language was developed to meet specific design requirements, making it the first language to be designed using well defined principles of software engineering.

The only tangible result of the design effort was a document -- the Reference Manual for the Ada Programming Language, normally referred to as the LRM. There was no compiler -- only the LRM, the language standard with which compilers must comply. Since the LRM was published, several Ada compilers have been developed, both under government contract and through funding by private industry. To date, the number of stands at seven, with new announcements occurring at an increasing pace.

The name Ada was officially applied to the new language in 1979. The name is not an acronym, as are COBOL and FORTRAN, but rather it is a proper name -- as is Pascal. The name was chosen to honor Ada Augusta, Countess of Lovelace (1815-1852), only daughter of the romantic poet Lord Byron and Lady Anabella Milbanke. Ada was a mathematician who, at an early age, became fascinated with Charles Babbage's difference engine -- so much so that she spent the rest of her short life involved in Babbage's work. She is recognized today as the world's first programmer.

The early proliferation of high order languages had a second, more subtle effect on software development. The greater the number of languages supported, the fewer software tools -- debuggers, full capability editors, etc. -- were developed, simply because of the dilution of resources. It was recognized early in the development of Ada that the adoption of a single high order language would make possible a standard support environment. A sequence of three documents outlining some technical and managerial issues of an Ada Programming Support Environment (APSE) were issued, under the following names

SANDMAN
PEBBLEMAN
STONEMAN

July, 1978
December, 1978
February, 1980:

The STONEMAN document² provides criteria for assessment and evaluation of APSE designs, and offers guidance for APSE designers and implementers.

The purpose of the APSE is to support the development of Ada software, and to support the maintenance of that software throughout its life-cycle. The three principle features of the APSE are:

- The data base, which contains all information associated with a software project throughout its life-cycle. This data refers to both technical and management issues.

- The interface, which includes a control language for access to the data base and to software tools in the toolset.

- The toolset, which includes software tools for program development, maintenance, and configuration control.

Also addressed by the STONEMAN requirements is the portability of user programs and software tools. By providing a standard support environment, the portability of a critical asset -- the programmer -- is enhanced.

In order to ensure the portability of Ada software, the Department of Defense is quite insistent that no subsets or extensions be permitted. This control over the language was accomplished by copyrighting the term Ada, which was done in 1981 with registration number 1,179,900. In December, 1980, the LRM was approved as a military standard -- MIL-STD-1815 -- and in February, 1983, after a massive coordination effort and some slight modification, it was approved as an ANSI standard. In order to meet the ANSI standard, or to qualify for permission to use the copyrighted term Ada, a compiler and its associated run-time support system must properly compile and execute a test suite of nearly two thousand separate tests. A compiler that satisfies this requirement is said to be validated. Compilers must be re-validated annually.

The Ada Joint Program Office (AJPO), within the Office of Undersecretary for Research and Engineering, U.S. Department of Defense, was established in December, 1980, and was charged with promotion of the Ada Language. Since that time, the AJPO has assumed several responsibilities, one of which is the establishment of the Ada

Validation Office, tasked with the validation of proposed Ada compilers.

After having spent six years in the development of a state-of-the-art, high order language that holds promise of vast economies in software life-cycle costs, the Department of Defense initiated a systematic approach to the promotion of the language. The establishment of the Ada Joint Program Office was the first step, but decisive action was required in order to bring the three services and their contractors into the process. A draft revision to DOD directive 5000.31 was promulgated in June, 1983:

Directive 5000.31 (revised) June 1983 -- requires all software for mission critical systems to be written in an authorized language, of which there are five: Ada, JOVIAL, FORTRAN, COBOL, and ATLAS. The Ada Language is specified as the single, common, computer programming language for Defense mission critical applications, effective January 1, 1984, for programs entering advanced development and July 1, 1984, for programs entering full scale engineering development. Waivers are permitted on a specific system or subsystem basis. COBOL is to be used only for business or administrative applications, but consideration of Ada for these applications is required where appropriate for machine-independence.

Mission critical computer resources -- referenced in the above directive -- are defined in accordance with 10 U.S.C. 2315 (the Warner amendment) and include computer resources used in the following:

- intelligence systems
- cryptologic systems
- military command and control systems
- weapon systems
- other applications critical to the direct fulfillment of military or intelligence missions.

Given these considerations, there should be no doubt about the intentions of the Department of Defense with respect to the future of the Ada language.

The infusion of a new, more capable programming language into the software

development process will not, in itself, solve the basic problem of poor quality software. What is required is the incorporation of modern software engineering concepts into the design itself; the Ada language was designed with these concepts in mind. It remains to develop methodologies that exploit the full capabilities of the language.

In late 1982, the Department of Defense announced its Software Initiative, a program intended to exploit the advantages of computer technology through software. The Software Initiative complements hardware initiatives within DOD, principally the Very High Speed Integrated Circuit (VHSIC) program -- which holds promise for greatly improved hardware performance through the use of sub-micron feature size semiconductor technology -- and the Defense Advanced Research Project Agency (DARPA) supercomputer project. Simply stated, the goal of the initiative is to improve software productivity while achieving greater system reliability and adaptability. To obtain its stated goal, the Software Initiative will focus on three major objectives:

- improving personnel resources
- increasing the power of software tools
- increasing the use of software tools.

The Ada language development represents a base upon which the initiative will be developed. Through the efforts of the initiative, the full capabilities of the language will be exploited.

Two other facets of the Software Initiative have emerged. In November, 1984, Carnegie-Mellon University was selected as the site of the Software Engineering Institute, established by the Department of Defense to serve as a focus for applied research and technology transfer within the initiative. One of the primary responsibilities of the institute will be the development and integration of workstations into the software development environment. There will be a significant emphasis on knowledge based artificial intelligence support systems, and on design methodologies.

The second development was the initiation of the Software Technology for Adaptable, Reliable Systems (STARS) program within the initiative. The STARS program will target improvement in the acquisition,

management, development, and support of software for military systems.

Ada has not been without its critics. Because the language design was a process open to public scrutiny, there was considerable criticism leveled against the design at various stages of development. Most of this criticism concerned the complexity of the language, and suggested a powerful subset with specially designed extensions. Many critics expressed their views when the ANSI standardization was undertaken, and again the question of subsets was predominant. Extensions and subsets, of course, were counter to the design goals of the language. In the end, the Ada language remained as a unique set of language requirements, with no subsets or extensions permitted.

Ada is considered by many to be the language of the future -- at least, of the near future -- not only for military computer systems, but also for commercial and industrial applications as well. It is a state-of-the-art, general purpose language that provides a rich set of capabilities for use in the application of modern software engineering techniques. It has the full support of the Department of Defense, both through the protection of the integrity of the design and through the support represented by the Software Initiative. It is a language tailored to the characteristics of modern hardware.

The Reference Manual for the Ada Programming Language may be obtained from the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C., 20402 as the ANSI/MIL-STD-1815A-1983.

Ada^R is a registered trademark of the U.S. Government (Ada Joint Program Office).

NEXT: The Characteristics of the Ada Language.

References:

1. "Reference Manual for the Ada Programming Language", ANSI/MIL-STD-1815A-1983. United States Department of Defense. 1983.
2. "Requirements for Ada Programming Support Environments", U.S. Department of Defense, "STONEMAN", February, 1980.

BASIC OS-9

by Ron Voigts

NO PERMISSION

I remember when I was a little tyke, I had to get permission from my mother to do something. Many times I would try something without asking and find I would get in trouble. My mother would scold me and tell me I had, "No PERMISSION!" So here I am 30 years later, from time-to-time when working with OS-9, I get "NO PERMISSION!" No, it's not mother. It's OS-9 letting me know that a file's attributes do not give me access to them.

I usually run into this problem when I get a new disk with some OS-9 software. I do a DIR on the new disk and see what's available. If everything is in upper case, I assume they are directories. So I DIR them too. Somewhere down the line I get the message:

ERROR #214

-NO PERMISSION

My experience has taught me that this must be a file. (By the way, it is best to put directories in upper case and leave files in lower case. It avoids confusion later on.)

All files and devices have attributes. Their attributes determine if they are a directory or a file. If they are shareable. And who has what kind of access. The attributes and their abbreviations are:

d	directory
s	shareable
pe	public execute permission
pr	public read permission
pw	public write permission
e	public execute permission
r	read permission
w	write permission

A handy command to use is ATTR, which is found in the execution directory. It has two variations, if you enter:

```
OS-9:attr my file
      r wr
```

it looks up the attributes for my file and reports them in the second line. They are reported in the same order I listed them above. This response means that I can read and write to my file. Another user on the system can read the file, but not write to it. The other way to use it is to follow the command line with changes to the file's attributes. The line:

```
OS9:attr my_file pw -w -r  
wr
```

adds public write permission, but removes my read and write permission. If you add a -a to the option list it will suppress printing out the files changed attributes. Using a -d, will demote a directory to a file. But not the other way around. If you want a directory, you'll have to use the MAKDIR command which makes new directories.

On a multi user system ATTR can be very useful. You can make files available for public use or restrict them for personal use. You may want to make a file only public read and so that anyone can read it, but still protect it from being altered. An accounting file may be given only public write, so employees can put data in, but not be able to read your business' finances.

What good is this for someone with only one terminal? At home if your entire family uses the system, everyone can have their own user ID number. They can change their files' attributes to however they see fit. Some files might be private and others be shared with the family. If you're the only user, you may still be interested in changing your files attributes. You could give a file only read access, to avoid inadvertently writing to it and destroying its contents.

Two other things should be mentioned. First even if you don't think you'll use ATTR, it is wise to leave it in your commands directory, /DO/CMOS. Some of the other commands may need it. DELDIR for one uses it. The other thing is when you purchase new software. Most companies are pretty good about setting up the attributes. But if you should have a problem running a new program, you might check its attributes.

LET A PROFESSIONAL HELP YOU SORT THINGS OUT

In last months column I talked about sorts. Creating a sort procedure can be fun. It can also be time consuming and discouraging. Consider what you have to do. You have to get the file. Then you must decide on some algorithm for sorting it. You have to consider memory and disk space. Another is its complexity. Do you want to sort on multiple keys? Plus are there any frills like adding up categories. Finally how do you want to write it back to the disk. Getting the results you want can be a programmer's nightmare.

The program presented last month was called "qsort". It would sort a file of up

to 100 lines, with the maximum line length of 80 characters. The sort used a quick sort method. I won't go into the actual workings of the sort. I will say that developing the method for the sort was a challenge. Even more of a challenge was debugging it. The sort worked fine until the files to be sorted got to be large. Then when I ran the sort, it would go off to "never-never land". After evenings of hard work I found the problem. Whenever two identical records were compared, the program would go into an endless loop trying to find the larger. Creating the sort took a lot of time. And yet by most standards, "qsort" was a simple sort.

If you want to sort files and don't want to spend a lot of time writing the sort routine, you might give thought to a "canned" program. The JBM Group has one package called SORTC. This is a sort that runs on Basic09 or Runb. While many sorts are limited to available memory, SORTC can use both memory and disk space, depending on the size of the sort. It has no restrictions on the amount of data to be sorted or the number of keys to be sorted. SORTC can also sum fields as it sorts.

Sorting on multiple keys makes it very useful. Some years ago I wrote a general ledger program. Each record in the program contained the date paid, the category, to whom it was paid and the amount. Being able to sort files for the ledger would have been very handy. For example, a file containing home budget information could first be sorted by category. This would be the main key. Records would be sorted according to food expenses, contributions to charities, medical payments and so forth. The second key could be to whom money was paid. Under the medical category payments to the doctor would be together, the dentist together and so on. A third key could be the date paid. Under doctor payments the dates would be sorted in ascending order. Now looking up a particular payment be fast and easy. First look up the category, then the name and finally the date. What could be faster and easier?

Another feature of SORTC is it can sum fields for a particular key as it sorts. Using our example from before, The main key can still be category and the second key, to whom paid. This time the sort sums the fields as it works. The result is a sum for each category-name in the file. Under the medical category, payments to the doctor would be totaled, payments to the

dentist totaled and so forth. It can also sum more than one field.

One thing should be mentioned, SORTC is a set of subroutines. You write the Basic09 procedure that uses SORTC to sort your files. This can have mixed blessings. It forces you to write the input/output routine for your sorts. You must also pass the records to and from SORTC by GOSUB's. On the positive side you have greater control over the sort. You can customize to your particular needs. You have contact with the data before it is sorted, so you may want to do something special with it.

One other thing SORTC is made to handle records of fixed length. In Basic09, these may have been created with the TYPE declaration. GET and PUT statements are used to read or write files. The problem with this is many times files are made up of records of varying lengths. It is easy to use the OS-9 editor to build files of simple lists. Such lists might include the members of your computer club, valuables in your house or perhaps some collection. I decided to attack this problem. The solution would be to input a line from the file, convert it to a record of fixed size and give it to SORTC. Later, after the sort, the record would be returned to as a simple line to the file. The listing at the end of the column show my solution.

The program is named simply enough, SORT. It expects two parameters to be passed to it. They are the source and the target files' names. The TYPE and DIM statements are used to create a complex variable, r.line. R.line is a string of 80 characters. The reason for using a complex variable instead of a standard string will soon be apparent. Next a path is opened to the source file and lines are input from it to r.line. The procedure PAD is used to add trailing spaces to the string. It does the opposite of the Basic09 command, TRIM\$. Next azu.charzu is set equal to r. This is interesting. R.line is a string. Azu.charzu is a complex variable of 80 bytes. It can't be set equal to r.line, but it can be set equal to r, which looks like 80 bytes. (By the way, this is one way to set unlike variables equal. Who said you can't mix apples with oranges.) A GOSUB 30 passes the string to SORTC.

A GOSUB 34 tells SORTC to do the sorting. The little ON ERROR routine

deletes leftover files from a previous sort. If the file doesn't exist an error occurs. The program goes on to line 100 and program execution goes on without a hitch.

The final part of the procedure creates the new, sorted file. R is set equal to bzu.charzu, which is the sorted record from SORTC. The record is written to the new file. A GOSUB to 25 gets another record. As long as funozu is not equal to 2, there are more records. When it does come back a 2, the sort is finished. We close the file and end.

The procedure PAD is another way in which variables can be changed. A string of 80 characters is passed to PAD. It receives it as a byte array of length 80. The WHILE...DO loop searches for the end of the string, which is 255 (that's a hex, \$FF). Starting at this point the remainder of the string is filled with spaces. (A space is hex \$20.) A routine like this good anytime you want to pad a string.

If you look at SORT in the listing, you're probably wondering where are the numbers for all those GOSUB's. They're added later. In fact if you write this program, when you exit it, it will probably give you half a dozen error messages. One message you'll get a lot is #074, that's an Undefined Line Number. Don't worry! JBM includes a program called SOTCGEN. You give it your specifications, like record size, the number of keys, do you want to sum fields and so forth. SOTCGEN then adds the necessary subroutines. Each line number is for a particular subroutine. Briefly they are:

```
LINE SUBROUTINE
10  Initialize sort
30  Pass records to sort
34  Start actual sort
25  Receive records from sort
```

After SOTCGEN is run, the procedure should be error free. If it isn't, it's "back to the drawing board" (or in our case the video terminal).

If you do have SORTC or are planning on getting, I think you'll like SORT. To use it, it is best to pack it first. Then enter

```
OS9:sort("file","sort_file")
```

You can use any names you like that are in your working directory. If they are not, be sure to use the full path name.

If writing your own sort routine still doesn't appeal to you, good news! The JBM Group has another solution. They also make a sort routine called GENUS. It is a stand-alone sort system. It doesn't need any other utilities or languages. (Remember SORTC needs a Basic09 environment to run.) It is written in C language and compiled into object code that can be called from your commands directory, /d0/cmds. GENUS does everything SORTC does. GENUS can also handle variable length records so you won't need a program like this month's program SORT.

If you are interested in sort routines you might consider these two sorts. If you want to write your own routines and add your own whistles and bells, you might consider SORTC. If you're looking for a good, all purpose sort routine at a reasonable price then GENUS is your sort.

Well that raps up another month. Take a look at the listing for SORT using the JBM subroutine SORTC. There are a lot of interesting features. It shows how to equate unlike variable types using the Basic09 commands TYPE and DIM. You might find other uses for procedure PAD. It's good for adding spaces to a variable string. Also start looking at your files' attributes. Not much thought is given to them, but they are a very important part of your files. Until next time, have fun!

```

PROCEDURE sort
(* this routine will sort a file *)
(* with line of varying lengths. *)

PARAM file 1, file 2:STRING[32]
TYPE record:line:STRING[80]
DIM r:record
DIM inpath,outpath:INTEGER

(* read the file to be sorted *)
OPEN #inpath, file 1:READ
GOSUB 10 \REM initialize workspace
WHILE NOT EOF(#inpath) DO
  READ #inpath,r:line
  RUN pad(r:line)
  @zu.charzu:=r
  GOSUB 30 \REM give record to sort
ENDWHILE
CLOSE #inpath

(* start jbm's sort routine *)
GOSUB 34

(* delete any old sorted files *)
ON ERROR GOTO 100
DELETE file 2
100 ON ERROR

(* we write an output file *)
CREATE #outpath, file 2:WRITE
WHILE funozu<>2 DO \REM 2 means eof
  r:=@zu.charzu
  r:line:=TRIMS(r:line)
  WRITE #outpath,r:line
  GOSUB 25 \REM get another line
ENDWHILE
END

```

```

PROCEDURE pad
(* this routine will add trailing *)
(* blanks to a string *)
PARAM s(80):BYTE
DIM i:INTEGER
DIM t:BOOLEAN

(* look for the end of the string *)
(* and add spaces if necessary *)
i:=1
t:=FALSE
WHILE i<=80 DO
  IF s(i)=255 THEN
    t:=TRUE
  ENDIF
  IF t=TRUE THEN
    s(i):=32
  ENDIF
  i:=i+1
ENDWHILE
END

```

CoCo User Notes

A Word From A Sponsor or Two

-- or --

by Carl Mann

A Fast Fling Around CoCo's

Garden of Delights

Let's make this column short and sweet. First some reviews, then some news, and then a little "how to use". Here goes...

First review: the **Uninterrupted Power Source** by Dynamic Electronics, Box 896, Hartselle, Alabama. 'Phone is (205) 773-2758.

The UPS is a dandy idea. Those of us who use low-end, high-performance micros in an industrial environment will especially appreciate what the UPS does for data reliability. As it happened, I had lost an entire section of a machine manual I was composing the day before the test unit arrived via (appropriately enough) UPS. I installed the device the very next morning.

The UPS consists of a 6 volt, 2.6 ampere-hour sealed rechargeable battery. It measures 5 1/4 inches wide by 2 5/8 inches high by 5/16 inches thick. A compact voltage and charge-rate regulator is attached to its "top" - that is, to the surface from which the terminals protrude. A cutout switch (to allow the user to power CoCo down) and a "system active" LED are also wired to the unit. Lead length is just right.

The UPS performed exactly as advertized in my old grey CoCo 1. I found that I could, for example, unplug the machine (with wordprocessor and text installed) for up to an hour and a half, plug the machine back in, and return to editing as though nothing had happened. It was wonderful! The battery fit under my "chiclet" keyboard just fine, too.

Ah, but that's the rub. I up and spent eighty-odd clams for a new keyboard. Guess what had to be removed to make room for the new keyboard? Right - the UPS. Darn. Then I dropped the new keyboard across the gap between the exposed 110-volt points on my faithful old beast's power supply and her CPU. (That'll teach me to follow my own rules - thank God my disk controller wasn't plugged in!) Blew every chip on the board. Had to drag out my reserve unit - a beige CoCo 2 with a different keyboard. Sure enough - the UPS wouldn't fit at all under the keyboard of the new unit. Tough luck. It was a bad day, anyway.

Ratings: In "D", "E", and TDP-100 CoCos, a 99. The unit is well-constructed and

should last as long as the machine it's installed in. It installs easily and fits (albeit tightly) under the keyboard as it should. CoCo 2 owners: for you, maybe 50. If you are clever and careful, the UPS could be mounted in a project box, and connected via a BNC or similar locking connector to your machine's power supply. The convenience of having a pre-engineered module to work with might just make the purchase worthwhile.

And now, gather 'round, friends. Riddle me this - what runs on a Color Computer, is a genuine creative tool, and is NOT a new wordprocessor or the latest megawhizbang programming language from the Zarneywhoop Group? Well, whatever you guessed, it's probably wrong - unless you guessed the Spectrum Adventure Generator from Spectrum Projects.

So what does an Adventure Generator do? Think of it as the link between your fingers and a universe of your own creation. The Adventure Generator enables you to compose an infinite variety of text adventure games (in MACHINE LANGUAGE, no less!) which may be saved to (and run from) either tape or disk. This is done through a number of BASIC source code generators and a machine-language runtime package which is united with the composer's code at the time of compilation.

The package includes the necessary tools with which both the traditional silent adventures and "talkies" may be composed. (Talkies require the Spectrum Voice Pack or similar device and a 64K machine.) Yes, you DO have to plan ahead and move very carefully while using the SAG system - but the documentation is quite clearly written for the non-programmer. (All you really need to know is how to minimally use a CoCo disk drive system.)

SAG enables the user to specify up to 70 verbs (or 400 bytes, whichever happens first) per adventure. (The average is said to be 50.) Up to 255 rooms (2800 bytes), 510 text screen responses to the player's input (8600 bytes), 255 objects (3100 bytes), unlimited Help Messages (well, "nearly" unlimited...), and 255 general purpose condition flags (Is the door locked? Is the gun loaded? Is the Creature from the Shocking Cesspool still as smelly as before?) may be had. Up to 700 program lines may be compiled per adventure. (8600 bytes for a silent one, 7680 for a "Talkie".)

Building an adventure has always been a complex process. SAG allows the composer to forget about the coding and therefore

to concentrate on the structure of the adventure itself. The manual strongly encourages the beginner to acquire systematic habits from the start. (A wordprocessor is suggested as an aid to creativity during this stage of the process. I agree. Whoever said you have to be a genius to do creative work on a keyboard was full of hawg jowls.) You must "lay out the scene" for the computer, define the objects to be found along the way, figure out what may (and may not) be done with them, when, where, and in what combination. You must define what words the player may use (and the computer may recognize). You still have to develop your style and make decisions. The difference is that with SAG on your side, that's ALL you have to do. No more struggling with tortuous if...then...gosub... etcetera in miles of labrynthine BASIC code. Instead, SAG provides you with a "room-by-room" listing (hardcopy only; printer required) of every location in the scenario and everything that is and can happen there. Clear, concise, and accurate. Who could ask for more?

I could - but you knew that already. The manual (which is pretty well printed, except for the sample Adventure listing, which was too wide for the paper) deserves better binding than the single staple in the corner. The system as delivered does NOT allow the compiled Adventure to be written to a secondary drive, thus requiring that it be written back to the system disk. (That's quite all right for single-drive users, of course, and the system files themselves are written in BASIC. I think (don't know) the whole thing could be redone for output to a second drive.)

All that notwithstanding, I rather like the Spectrum Adventure Generator. It appears to be bug-free and well thought out. It certainly is a laborsaver as well as a genuine creative tool. I'd rate it about 90 overall. It could grow on me. I don't think I'll tire of it very soon.

By the way, neither does Spectrum Projects. In fact, the company sells a one-time license (for \$50.00) which entitles the SAG user to market a completed creation commercially. (Spectrum will evaluate your product for saleability, of course!) I wonder what sort of a deal I could cut with Doug Addams to adventurize his "Hitch-hiker's Guide to the Galaxy" series? It's the perfect raw material for this sort of thing, and I won't be miffed if someone else beats me to it... Ask your

local library and/or bookstore. You'll see what I mean. Tally ho!

Next item: a word about Star Kits' STAR-DOS PLUS (available from Southeast Media - see Adv. this Issue) for the Color Computer. I've been working my old grey CoCo out with an advance copy, and I am genuinely impressed. It is a real, live, whizbang of a DOS. It appears to run everything that runs under FLEX (for example, STYLOGRAPH is right at home) with ease. It allows any mix of 40, 80, and Hard Disk drives up to the capacity of your System. (Up to 255 tracks may be formatted on BOTH sides of the platter, assuming your drive has that many!) It comes with a full complement of file-handling and maintenance utilities. I haven't used it much yet (have to buy a bunch of disks and a new disk tub before proceeding) but I must say this: STAR-DOS PLUS may in fact be the best non-ROM Disk Operating System ever to be run on a CoCo. It's compact, powerful, and extremely flexible. I'll keep you posted.

Last item: Spell N' Fix 2 for the Color Computer, also from Star-Kits. I've given the 40,000 word version a real workout this month. I tried to use VIP Speller to proofread the text of a technical manual I had written, only to find out that the dictionary didn't really have the "technical background" that would qualify it for the job. Spell N' Fix 2, on the other hand, does. VIP Speller failed to recognize so many common technical words, I wound up telling it to ignore MIS-SPELLED words out of sheer reflex! Then I tried Spell N' Fix 2. The difference was amazing. Its memory-resident (and easily-edited) 200 word "core dictionary" is machine-language fast. The 40,000 word main dictionary is well-compressed, and fits on a 35-track disk with room to spare. All 40,000 words appear to be exceptionally well-chosen. The result? I caught and fixed MORE ERRONEOUS WORDS with Spell N' Fix 2.

Spell N' Fix 2 is available for free (YES! FREE!! REALLY!!!) from Star-Kits. (See the ad in this issue.) You get a lot more than just the simple spelling checker, though. You actually get TWO versions of the program. Spell N' Fix 1 is included because it contains advanced features that will be of real interest to the serious programmer. A "bit image manual" (in Telewriter ASCII text format) is included on the disk, as is the 20,000 word original

Spell N' Fix dictionary that was making 'em drop their dentures long before there was anything else available at all. Just send in a disk to receive it all.

Well, almost all. Notice that there seems to be a catch. Actually a few catches. First, there is a great deal more to Spell N' Fix than can be put into a brief, print-it-yourself manual. (The manual-on-a-disk tells you as much.) Second, the "freeware" version only includes the 20,000 word dictionary. Third, the program asks you to send a contribution to the author every time you boot it up. Fourth, the full manual (very well printed and bound with a lie-flat GBC spine) and the other 20,000 words will cost you at least fifty bucks. Fifth, there IS a way to make Spell N' Fix 2 stop asking you for money. (It even will occasionally do so in the middle of a spell-checking run!) It's in the fifty-dollar manual. And I'm NOT telling! Why? Simple. Spell N' Fix is the best darn spelling checker on the market for the CoCo - and I've tried them all. (The Shack's idea of a spellchecker is so primitive I hesitate to even mention it.) Spell N' Fix 2 is very easily worth the \$50.00 asking price for the 20,000 word version alone. \$75 to \$100.00 would be more in line with its features and performance.

Comparable products running on schizoid Big Blue machines (and their brain-damaged clones) cost twice as much, and don't allow half as much editing and customization as Spell N' Fix 2. I tell you, we've got it better than we think. (If you don't believe me, go shopping for computers. You'll come back home to CoCo real fast.) Spell N' Fix 2 rates 100+ in my little gray book. Try it. You'll come back for more, and be glad you did.

Tip of the month. Next time you have lots of number-crunching to do, insert an EXEC 320 into your BASIC number-crunching program instead of scotch-taping that silly little piece of cardboard to the keyboard. You know - the one that says, "DATA PROCESSING IN PROGRESS! DO NOT TOUCH!" in Magic Marker across its face, right? The BASIC program will still run, but the keyboard will be completely disabled. (Press <RESET> to recover.) I don't know why it works, but work it does.

It's late at night, and there's some distance to drive before I sleep. Gotta go! Until next month...

Using FLEX/Star-DOS

by Troy Brumley

I'd like to welcome all of the ex-Color Micro Journal readers to '68' Micro Journal. I imagine that most of you are a little confused about the non-CoCo coverage that fills the rest of this magazine. Don't panic! It is all easy to understand, and most of it can be useful to you if you have a disk system with either FLEX, STAR-DOS, or OS-9.

If you don't already have one of those operating systems, you may want to buy one. In this series of articles I hope to provide enough information about FLEX and STAR-DOS to help new FLEX/STAR-DOS owners learn how to use their systems, and also to help prospective buyers of those systems decide if they will enjoy using them. Ron Voigts is writing a column on BASIC OS-9, so I will concentrate primarily on FLEX and STAR-DOS.

A NOTE ABOUT STAR-DOS

STAR-DOS is a FLEX compatible operating system marketed by STAR-KITS. It was developed with an eye towards being able to use existing applications software while providing some improvements over FLEX. Any FLEX software you buy will also run on STAR-DOS. From here on in I will refer only to FLEX since that is the system I use, but anything I say applies equally to STAR-DOS, other than possibly some minor syntax differences.

WHAT IS FLEX?

FLEX is an Advanced Operating System (AOS). It provides a standard environment for 6800 and 6809 based software to run in. This standard environment includes a well documented set of system calls that will allow programs written using them to run on a variety of 68xx based computer systems.

These standard system calls perform the same function (open a disk file, display a character, report an error, etc) on all FLEX systems. While each computer may use different input/output devices your FLEX based software doesn't care, since FLEX works the same on all 6809 systems. All you (or your FLEX supplier) needs to do to get FLEX running on a new 6809 based machine (like the CoCo) is to write a few input/output drivers and patch them into FLEX.

The only requirements for running FLEX on a CoCo is 64K of RAM and at least a

drive 0. Most any disk system will work with FLEX since it uses no DISK BASIC ROM routines once it is booted. I use a J&M controller with JDOS 1.11. I have experienced no problems using FLEX with this system.

WHAT GOOD DOES THIS DO ME?

In plain language this means that you can probably run any of a number of excellent programs written for 'big' 6809 systems (GIMIX, SWTPC, HELIX, etc) on your CoCo. This opens up a whole new world of options for the CoCo owner. Examples of software available for the CoCo running FLEX include: the DYNACALC and TABULA RASA spread sheet programs, the STYLOGRAPH, SCREDITOR III, and DYNASTAR/DYNAFORM word processors, at least 2 PASCAL compilers, 2 or three C compilers, COBOL, BASIC, FORTH, 3 spelling checkers, disk sort/merge utilities, sophisticated terminal programs, business accounting packages, payroll systems, and so on.

Of course this software is not free. FLEX can be purchased for prices ranging from (roughly) \$50.00 up to \$150.00. Each implementation has its good and bad points. The F-MATE conversion sold by DATACOMP is the first (and, according to several independent CoCo software & hardware firms that I've talked to, the best) version of FLEX for the CoCo.

A TALE OF TWO FLEXES

There are three or four versions of FLEX for the CoCo, but only two are in widespread usage. These are the F-MATE conversion from DATACOMP and Frank Hogg Labs (FHL) FLEX. Both cost roughly the same and seem to be liked by their owners.

Unfortunately, while FLEX is supposed to be a STANDARD operating system, there are a few hardware dependent differences that can cause a few problems for existing FLEX software. Fortunately many FLEX software authors have special versions of their software available for popular systems (such as the CoCo).

The only differences I know of between F-MATE and FHL FLEX are the video drivers for the hi-res screen display and double sided disk formatting. The only types of software that bypass the normal FLEX input/output routines for the screen and keyboard are full screen oriented programs such as Word Processors and Spread Sheet programs. I am writing this article using STYLOGRAPH from Stylo Software Inc. on a CoCo using the F-MATE FLEX system. STYLO

came with video drivers for both FHL and F-MATE FLEX so I would have no problems using it with either system. If you aren't sure if a piece of FLEX based software will run on your CoCo, feel free to ask your supplier.

The incompatibility of double sided disk formats should cause no problems since a "standard" FLEX disk is single sided single density with 35 tracks. Any special CoCo versions will be sold on single sided double density 35 track disks since that is the "standard" FLEX/STAR-DOS format. Double sided disks are a user option and don't really bother software once it is running on your computer, but you may have some trouble sharing data files with friends that have a different FLEX conversion if you do have double sided disks.

WHY WOULD I WANT FLEX?

There are a few reasons why you might wish to buy FLEX, STAR-DOS, or even OS-9. The most obvious one is that you want to play with an AOS. I suspect that both FLEX and OS-9 can be equally fun to play around with.

You may want to run some serious business software such as a payroll or general ledger system. There are several such systems available for FLEX. OS-9 is still a young operating system and it won't have too much business software available for at least another year or two (although **K-BASIC** provides a means of running the FLEX-Based BASIC Software under OS-9).

You might want to learn another programming language. Both FLEX and OS-9 have several languages available. I would choose FLEX over OS-9 in this situation simply because FLEX runs with no trouble on a single drive system. Unless you need, or want, to learn Basic09, I would lean towards FLEX.

OS-9 is a UNIX-like operating system. Every three months or so one of the other computer magazines (hobby or business) devotes an issue or two to UNIX and the C programming language. Somebody out there feels that UNIX is the wave of the future. If you agree with them you should get OS-9.

WHAT'S NEXT

Next month we'll start working with FLEX in earnest. I'll explain how I installed FLEX on my system and give examples of just how powerful FLEX is.

Until then...

The Apple Macintosh

by Robert L. Nay

The power of the 68000 is amply demonstrated by its overwhelming domination of the larger CAD/CAM and UNIX-Oriented Computer Systems, but there has been very little available for the individual User that would allow him to "get in on the Ground Floor" with a 68000-Based System in the lower priced Personal Computer System range. In the good-old-days, the hobbyist/experimenters would have gathered around an inexpensive Single Board Computer System and, in the process of working with it, would have developed an Operating System and a bunch of Software to go with it. Sadly, this did not happen when the 68000 first appeared; possibly because the Personal Computer Industry has matured to the point that there are few pioneers left, or maybe due to the low number of inexpensive 68000-SBC's that were produced, or, probably, due to a combination of both factors. Our tastes now require at least a Disk System, and a decent Graphics capability is becoming a "Standard" requirement.

There was a lot of initial interest in the Sinclair QL Computer, as it was one of the first to offer a "minimally acceptable" System, but it has been delayed to the point that most of that interest has dissipated, and it may find it hard to carve a niche in the Industry. Atari is making noises about a \$1000 68000-Based, 512K, Color Computer System which is now scheduled for release in June, and there are rumors of other similar Systems that MIGHT show up in the next couple of years. But at this time, Apple appears to be in the best position to nab a large segment of the 68000-Based low-priced Computer Market with their "Apple 32" Product Line, which was introduced last year.

Several factors make the Apple 68000 Systems attractive. Apple is a real "Market Force" in the Personal Computer arena, and is attempting to become one in the Business area. They are large enough to generate a "Standard" all by themselves (witness the IBM PC Industry scramble to copy the Apple-initiated Windows User Interface, or the MacPaint copies). The new Atari System will have a User Interface similar to the Mac, and it has been hinted that it will not be hard to port Macintosh Application Software over to the Atari Computer. With Apple presently gearing up

their production of Macs to 100,000 per month, a potential Mac Purchaser is assured that there will be Macintosh's around for a while, and that there will be plenty of support for his investment.

The major products in the "Apple 32" Product Line presently include the Macintosh Computer Systems (the Apple Lisa 2 and Lisa 2/5 are being phased out, and the Lisa 2/10 has been renamed the Macintosh XL), the recently released LaserWriter Laser Printer, the yet to be released FileServer Network Controller/File Management System, and numerous peripherals that go with the overall System. In addition, Apple has also announced their AppleTalk Networking System, which allows up to 32 Macs, LaserWriters, ImageWriters, etc., to be interconnected at a cost of around \$50 per connection to the Network.

Does the Macintosh offer "the" Solution? Obviously not; no single Computer System will fill every need (in spite of IBM's attempts to the contrary). Maybe the Macintosh "User Interface" is not the ideal General-User interface, but our experience here indicates that it is MAGNITUDES better than a powerful "Unix-like" interface for the vast majority of the Computer Users in the world who have no concept of a "bit" or "byte", or of a "sector", or a "path", "field", "record", and on and on. These "Computer Users" want a Computer to SOLVE their problems, not create more problems by having to learn how to "think like a Computer" to use the Operating System, or learn a Programming Language, or be able to use a Data Base Management package. THEY need a Computer that "thinks like THEY do", not one that thinks like a COMPUTER!

The Mac Interface may be a little frustrating to those of us that understand Computers and LIKE the power and flexibility of something like FLEX/STARDOS or the Unix-oriented Systems such as OS-9, but most Computer Users need something to allow them to get an application up-and-running with as little computerese as possible. Why should a Business Computer System in a small office require a "Computer Support Section"? Why can't Retailers in general sell a simple Computer System to a Business Customer without requiring MANY HOURS of explanation about how to do this, or why you have to do that before you can do this, etc.? Why should a Retailer have to pass up a potential Sale just because he knows that that customer

would not be able to use the Operating System on the Computers that he sells; or because there is not a "canned" solution to their problem (even though he knows that his Computer System could do the job IF they would take the time to learn the "System" and how to use the Software that IS available for it)?

Apple's concept with the Macintosh Computer SYSTEM is to provide a System that "thinks" like a normal user, and a System that can expand as the User's needs grow. Rather than provide a "Smart, SUPER Computer" that will do anything that a User might ever want to do with a Computer System, they are making each unit of the System "stand on its own". If a User does not need a Laser Printer, why should he have to purchase a Computer that has the Memory and Power to develop a 300 dot/in. drive for that Printer; why not let the PRINTER do that, and put the "smarts" in IT? Since most "Office Units" consist of 5 to 25 people working on approximately the same thing, why not have a simple, inexpensive "Network" for up to 32 "high-performance peripherals", each doing its own thing? The Macs would provide the "User Interface" to the information in the Network; the LaserWriter could be the Network's "System Printer". A "smart File Server" with a 20 to 40 Meg Hard Disk could handle such things as communicating with Mainframe Networks and tying other 32-Unit Networks together, provide file transfers, electronic mail, print spooling, file management, and process multiuser applications when the User needed these capabilities. The whole Apple "Office System" concept is built on a "pay as you go" foundation.

There seems to be a prevailing impression that there is a lack of Software Support for the Macintosh. The immediate reply to this statement is; relative to WHAT? There is already several times as many Programs available for the Macintosh as there are for the whole SS-50 Bus community. There are more, and MUCH more useful, Programs available for the Mac now than there was for the Color Computer, for example, a year after IT was released. But, obviously, there is not as much available NOW for the Mac as there is for the IBM PC, or the UNIX environment.

Apple states that there are now "more than 300 Software Packages" available for the Mac. There are a few holes as of today (for example, there is next to no



FINALLY !!

Now you can run

TSC XBASIC Programs Compiled to Asmb. Lang.,
under **OS-9™**, **CoCo OS-9**, or **FLEX™** with

★ **K-BASIC** under **OS-9** and **FLEX** will now compile
TSC BASIC, XBASIC, and XPC Source Code Files

K-BASIC now makes the multitude of **TSC MBASIC** Software
available for use under **OS-9**. Transfer your favorite **BASIC**
Programs to **OS-9**, compile them, Assemble them, and
FINISH -- usable, multi-precision, familiar Software is
running under your favorite Operating System!

★ **K-BASIC** (**OS-9** or **FLEX**), including the **OSM Assembler**
\$199.00

Limited time SPECIAL offer!

South East Media and Lloyd I/O is offering OS-9

★ **K-BASIC** Compiler w/**OSM Assembler** AND
O-F FLEX to **OS-9** Transfer Program
BOTH for **\$199.95**

Offer **ONLY** valid when ordered as a **Package** from **South East Media**
=====



Basic09 Tour Guide



by Dale Puckett -- An excellent Book on using **OS-9**. Oriented
towards using the powerful **Basic09 Programming Language**. It also
contains a lot of good information on using **OS-9** in general.

Normally \$18.95
Special ---- NOW only \$16.00



== SHIPPING ==
Add 2X U.S.A.
(min. \$2.50)
Add 5X Surface Foreign
10X Air Foreign

*FLEX is a trademark of Technical Systems Consultants
*OS9 is a trademark of Microware



Availability Legend ---
P = FLEX, CCP = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

FREE DISKETTE WITH EVERY \$50 PURCHASE

TOLL FREE TELEX 550 414 PVT BTH

1-800-338-6800
For Ordering

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE



--- FLEX Software ---



TSC "Flex Utilities"	was \$75.00,	NOW only \$65.00
TSC "Sort Merge"	was \$75.00,	NOW only \$65.00
TSC "6809 Basic"	was \$75.00,	NOW only \$65.00
TSC "Extended Basic"	was \$100.00,	NOW only \$90.00
TSC "DeBug"	was \$75.00,	NOW only \$65.00
TSC "FLEX Diagnostics"	was \$75.00,	NOW only \$65.00
TSC "Text Processing System"	was \$75.00,	NOW only \$65.00
TSC "68000 Cross Assembler"	was \$250.00,	NOW only \$199.95

!!! Please Specify Your Operating System & Disk Size !!!

FREE DISKETTE WITH EVERY \$50 PURCHASE

TOLL FREE 1-800-338-6800
For Ordering

TELEX 550 414 PVT BTM

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
Hixson, TN 37343

for information
call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE



ASSEMBLERS

ASTRUK09 from Southeast Media -- A "Structured Assembler for the 6809" which requires the TSC Macro Assembler. F, CCF - \$99.95

Macro Assembler for TSC -- The FLEX STANDARD Assembler.
Special -- CCF \$35.00; F \$50.00

OSM Extended 6809 Macro Assembler from Lloyd I/O. -- Provides local labels, Motorola S-records, and Intel Hex records. Can generate OS-9 Memory modules under FLEX. FLEX, CCF, OS-9 \$99.00

Relocating Assembler w/Linking Loader from TSC. -- Use with many of the C and Pascal Compilers. F, CCF \$150.00

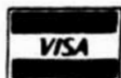
MACE, by Graham Trott from Windrush Micro Systems -- Co-Resident Editor and Assembler; fast interactive A.L. Programming for small to medium-sized Programs. F, CCF - \$98.00

TRUE CROSS ASSEMBLERS from Computer Systems Consultants -- Supports 1802/5, Z-80, 6800/1/2/3/8/11/MC11, 6804, 6805/MC05/146805, 6809/00/01, 6502 family, 8080/5, 8020/1/2/35/C35/39/40/48/C48/49/C49/50/8748/49, 8031/51/8751, and 68000 Systems. Assembler and Listing formats same as target CPU's format. Produces machine independent Motorola S-Text.
FLEX, CCF, OS-9, UniFLEX each - \$50.00
any 3 - \$100.00
the complete set (including the C Source) - \$200.00

XASM Cross Assemblers for FLEX from Compusense Ltd. -- This set of 6800/1/2/3/5/8, 6301, 6502, 8080/5, and Z80 Cross Assemblers uses the familiar TSC Macro Assembler Command Line and Source Code format, Assembler options, etc., in providing code for the target CPU's. Complete set, FLEX only - \$150.00

CRASMB from Lloyd I/O -- 8-Bit Macro Cross Assembler with same features as OSM, generates OSM Macros which cross-assemble to 6800/1/2/3/4/5/8/9/11, 6502, 1802, 8048/80/85, Z-80. Supports the target chip's standard mnemonics and addressing modes.
FLEX, CCF, OS-9 Full package -- \$399.00

CRASD 16.32 from Lloyd I/O -- Cross Assembler for the 68000.
FLEX, CCF, OS-9 \$249.00



== SHIPPING ==
Add 2X U.S.A.
(min. \$2.50)
Add 3X Surface Foreign
10X Air Foreign

TOLL FREE 1-800-338-6800
For Ordering

SOUTH EAST MEDIA

5900 Cassandra Smith Rd. CoCo OS-9™ FLEX™
Hixson, TN 37343
info (615) 842-4601

SOFTWARE

*FLEX is a trademark of Technical Systems Consultants
**OS9 is a trademark of Microware

DISASSEMBLERS

SUPER SLEUTH from Computer Systems Consultants -- Interactive Disassembler; extremely POWERFUL Disk File Binary/ASCII Examine/Change, Absolute or FULL Disassembly. XREF Generator, Label "Name Changer", and Files of "Standard Label Names" for different Operating Systems

Color Computer 55-50 Bus (all w/ A.L. Source)
CCO (32K Req'd) Obj. Only \$49.00 F, \$99.00
CCF, Obj. Only \$50.00 U, \$100.00
CCF, w/Source \$99.00 O, \$101.00
CCO, Obj. Only \$50.00

DYNAMITE + from Computer Systems Center -- Excellent standard "Batch Mode" Disassembler. Includes XREF Generator and "Standard Label" Files. Special OS-9 options w/ OS-9 Version.

CCF, Obj. Only \$100.00 CCO, Obj. Only \$59.95
F, " " \$100.00 O, " " \$150.00
U, " " \$300.00

PROGRAMMING LANGUAGES

PL/9 from Windrush Micro Systems -- By Graham Trott. A combination Editor/Compiler/Debugger. The Single-Pass Compiler supports large Symbol Names; Variable Types; Pointers; Control Structures; Stack, A-, B-, and D-Register manipulation; etc. Includes Source-Oriented Debugger. F, CCF - \$198.00

MINISCAL from Whimsical Developments -- Now supports Real Numbers. "Structured Programming" WITHOUT losing the Speed and Control of Assembly Language! Single-pass Compiler features unified, user-defined I/O; produces ROMable Code; Procedures and Modules (including pre-compiled Modules); many "Types" up to 32 bit Integers, 6-digit Real Numbers, unlimited sized Arrays (vectors only); Interrupt handling; long Variable Names; Variable Initialization; Include directive; Conditional compiling; direct Code Insertion; control of the Stack Pointer; etc. Run-Time subroutines inserted as called during compilation. Normally produces 10% less code than PL/9. F and CCF - \$195.00

C Compiler from Windrush Micro Systems by James McCosh. Full C for FLEX except bit-fields, including an Assembler. Requires the TSC Relocating Assembler if user desires to implement his own Libraries. F and CCF - \$295.00

C Compiler from Introl -- Full C except Doubles and Bit Fields, streamlined for the 6809. Reliable Compiler; FAST, efficient Code. More UNIX Compatible than most. F, CCF, and O - \$375.00 U - \$425.00

PASCAL Compiler from Lucidata -- ISO Based P-Code Compiler. Designed especially for Microcomputer Systems. Allows linkage to Assembler Code for maximum flexibility. F and CCF 5" - \$190.00 F 8" - \$205.00

PASCAL Compiler from OmegaSoft -- For the PROFESSIONAL; ISO Based, Native Code Compiler. Primarily for Real-Time and Process Control applications. Powerful; Flexible. Requires a "Motorola Compatible" Relocating Asmb. and Linking Loader. F and CCF - \$425.00 One Year Maint. - \$100.00

★ **Z-BASIC** from LLOYD I/O -- A "Native Code" BASIC Compiler which is now fully TSC ZBASIC compatible. The single-pass compiler compiles to Assembly Language Source Code (which may be assembled by the included OSM Assembler, or by the CRASD Cross Assemblers). Conditional assembly reduces Run-time package. FLEX, CCF, OS-9 Compiler with OSM Assembler - \$199.00

CRUNCH COBOL from Compusense Ltd. -- Supports large subset of ANSI Level 1 COBOL with many of the useful Level 2 features. Full FLEX File Structures, including Random Files and the ability to process Keyed Files. Segment and link large programs at runtime, or implemented as a set of overlays. The System requires 56K and CAN be run with a single Disk System. FLEX, CCF; Normally \$199.00
Special introductory Price (while in effect) -- \$99.95

FORTH from Stearns Electronics -- A CoCo FORTH Programming Language. Tailored to the CoCo! Supplied on Tape, transferable to disk. Written in FAST ML. Many CoCo functions (Graphics, Sound, etc.). Includes an Editor, Trace, etc. Provides CPU Carry Flag accessibility, Fast Task Multiplexing, Clean Interrupt Handling, etc. for the "Pro". Excellent "Learning" tool!
Color Computer ONLY - \$58.95

Availability Legend

F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCO = Color Computer Disk
CCF = Color Computer Tape

!!! Please Specify Your Operating System & Disk Size !!!



SOFTWARE DEVELOPMENT

Basic09 XRef from Southeast Media -- This Basic09 Cross Reference Utility is a Basic09 Program which will produce a "pretty printed" listing with each line numbered, followed by a complete cross referenced listing of all variables, external procedures, and line numbers called. Also includes a **Program List Utility** which outputs a fast "pretty printed" listing with line numbers. Requires Basic09 or Run8.

Q & CCO obj. only -- \$39.95; w/ Source -- \$79.95

Lucidata PASCAL UTILITIES (Requires LUCIDATA Pascal ver 3)

XREF -- produce a Cross Reference Listing of any text; oriented to Pascal Source. F and CCF - \$25.00

INCLUDE -- Include other Files in a Source Text, including Binary; unlimited nesting capabilities. F and CCF - \$25.00

PROFILER -- provides an indented, numbered, "Structogram" of a Pascal Source Text File; view the overall structure of large programs, program integrity, etc. Supplied in Pascal Source Code; requires compilation. F and CCF - \$25.00

DUB from Southeast Media -- A UnifLEX "basic" De-Compiler. Re-Create a Source Listing from UnifLEX Compiled basic Programs. Works w/ ALL Versions of 6809 UnifLEX basic. U - \$219.95

FULL SCREEN FORMS DISPLAY from Computer Systems Consultants -- TSC Extended BASIC program supports any Serial Terminal with Cursor Control or Memory-Mapped Video Displays; substantially extends the capabilities of the Program Designer by providing a table-driven method of describing and using Full Screen Displays. F and CCF - \$50.00, U - \$75.00

DISK UTILITIES

OS-9 Disk from Southeast Media -- For Level I only. Use the Extended Memory capability of your SWTPC or Gmix CPU card (or similar format DAT) for FAST Program Compiles, CMD execution, high speed inter-process communications (without pipe buffers), etc. - SAVE that System Memory. Virtual Disk size is variable in 4K increments up to 660K. Some Assembly Required. -- Level I ONLY -- OS-9 obj. only -- \$79.95; w/ Source -- \$149.95

O-F from Southeast Media -- Written in BASIC09 (with Source), includes: REFORMAT, a BASIC09 Program that reformats a chosen amount of an OS-9 disk to FLEX Format so it can be used normally by FLEX; and FLEX, a BASIC09 Program that does the actual read or write function to the special O-F Transfer Disk; user-friendly menu driven. Read the FLEX Directory, Delete FLEX Files, Copy both directions, etc. FLEX users use the special disk just like any other FLEX disk. O - \$79.95

COPYMULT from Southeast Media -- Copy LARGE Disks to several smaller disks. FLEX utilities allow the backup of ANY size disk to any SMALLER size diskettes (Hard Disk to floppies, 8" to 5", etc.) by simply inserting diskettes as requested by COPYMULT. No fooling with directory deletions, etc. COPYMULT.CMD understands normal "copy" syntax and keeps up with files copied by maintaining directories for both host and receiving disk system. Also includes BACKUP.CMD to download any size "random" type file; RESTORE.CMD to restructure copied "random" files for copying, or recovering back to the host system; and FREELINK.CMD as a "bonus" utility that "relinks" the free chain of floppy or hard disk, eliminating fragmentation.

Completely documented Assembly Language Source files included. ALL 4 Programs (FLEX, 8" or 5") \$99.50

COPYCAT from Lucidata -- Pascal NOT required. Allows reading TSC Mini-FLEX, SSB DOS68, and Digital Research CP/M Disks while operating under FLEX 1.0, FLEX 2.0, or FLEX 9.0 with 6800 or 6809 Systems. COPYCAT will not perform miracles, but, between the program and the manual, you stand a good chance of accomplishing a transfer. Also includes some Utilities to help out. Programs supplied in Modular Source Code (Assembly Language) to help solve unusual problems. F and CCF 5" - \$50.00 F 8" - \$65.00



== SHIPPING ==

Add 1% U.S.A.
(min. \$2.50)
Add 5% Surface Foreign
10% Air Foreign

*FLEX is a trademark of Technical Systems Consultants
**OS9 is a trademark of Microware

TOLL FREE
1-800-338-6800
For Ordering

5900 Cassandra Smith Rd.
Hixson, TN 37343
info (615) 842-4601

South East Media
CoCo OS-9™ FLEX™
SOFTWARE

FREE DISKETTE WITH EVERY \$50 PURCHASE

TOLL FREE
1-800-338-6800
For Ordering

TELEX 558 414 PVT BTH

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
Hixson, TN 37343

for information
call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE

FLEX DISK UTILITIES from Computer Systems Consultants -- Eight (8) different Assembly Language (w/ Source Code) FLEX Utilities for every FLEX Users Toolbox: Copy a File with CRC Errors; Test Disk for errors; Compare Two Disks; a fast Disk Backup Program; Edit Disk Sectors; Linearize Free-Chains on the Disk; print Disk Identification; and Sort and Replace the Disk Directory (in sorted order). -- PLUS -- Ten XBASIC Programs including: A BASIC Reorganizer with EXTRAs over "RENUM" like check for missing label definitions, processes Disk to Disk instead of in Memory, etc. Other programs Compare, Merge, or Generate Updates between two BASIC Programs, check BASIC Sequence Numbers, compare two unsequenced files, and 5 Programs for establishing a Master Directory of several Disks, and sorting, selecting, updating, and printing paginated listings of these files. A BASIC Cross-Reference Program, written in Assembly Language, which provides an X-Ref Listing of the Variables and Reserved Words in TSC BASIC, XBASIC, and PRECOMPILER BASIC Programs. ALL Utilities include Source (either BASIC or A.L. Source Code). F and CCF - \$50.00

COMMUNICATIONS

CHODEN Telecommunications Program from Computer Systems Consultants, Inc. -- Menu-Driven; supports Dumb-Terminal Mode, Upload and Download in non-protocol mode, and the CP/M "Modem" Christensen protocol mode to enable communication capabilities for almost any requirement. Written in "C". FLEX, CCF, OS-9, UnifLEX; with complete Source -- \$100.00 without Source -- \$50.00

XDATA from Southeast Media -- A COMMUNICATION Package for the UnifLEX Operating System. Use with CP/M, Main Frames, other UnifLEX Systems, etc. Verifies Transmission using checksum or CRC; Re-Transmits bad blocks, etc. U - \$299.99

GAME

RAPIER - 6809 Chess Program from Southeast Media -- Requires FLEX and Displays on Any Type Terminal. Features: Four levels of play. Swap side. Point scoring system. Two display boards. Change skill level. Solve Checkmate problems in 1-2-3-4 moves. Make move and swap sides. Play white or black. This is one of the strongest CHESS programs running on any microcomputer, estimated USCF Rating 1600+ (better than most 'club' players at higher levels). F and CCF - \$79.95

Availability Legend --

F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UnifLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

!!! Please Specify Your Operating System & Disk Size !!!

FREE DISKETTE WITH EVERY \$50 PURCHASE

TOLL FREE **TELEX 558 414 PVT 8TH**
1-800-338-6800
 For Ordering

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
 Hixson, TN 37343
 for information
 call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE



WORD PROCESSING

SCHEIDT III from Windrush Micro Systems -- Powerful Screen-Oriented Editor/Word Processor. Almost 50 different commands; over 300 pages of Documentation with Tutorial. Features Multi-Column display and editing, "decimal align" columns (AND add them up automatically), multiple keystroke macros, even/odd page headers and footers, imbedded printer control codes, all justifications, "help" support, store common command series on disk, etc. Use supplied "set-ups", or remap the keyboard to your needs. Except for proportional printing, this package will DO IT ALL!

6800 or 6809 FLEX or SSB DOS, OS-9 - \$175.00

STYLOGRAPH from Great Plains Computer Co. -- A full-screen oriented WORD PROCESSOR -- (uses the 51 x 24 Display Screens on CoCo FLEX/STAR-DOS, or PBJ Wordpak). Full screen display and editing; supports the Daisy Wheel proportional printers.

SPECIAL CCF and CCO - \$99.95, F or D - \$295.00, U - \$395.00

SPELL from Great Plains Computer Co. -- Fast Computer Dictionary. Complements Stylograph.

SPECIAL CCF and CCO - \$69.95, F or D - \$125.00, U - \$175.00

MAIL MERGE from Great Plains Computer Co. -- Merge Mailing List to "Form" Letters, Print multiple Files, etc., through Stylo.

SPECIAL CCF and CCO - \$59.95, F or D - \$145.00, U - \$175.00

JUST from Southeast Media -- Text Formatter developed by Ron Anderson; for Dot Matrix Printers, provides many unique features. Output "Formatted" Text to the Display. Use the FPRINT.COM supplied for producing multiple copies of the "Formatted" Text on the Printer INCLUDING IMBEDDED PRINTER COMMANDS (very useful at other times also, and worth the price of the program by itself). "User Configurable" for adapting to other Printers (comes set up for Epson MX-80 with Graftrax); up to ten (10) Imbedded "Printer Control Commands". Compensates for a "Double Width" printed line. Includes the normal line width, margin, indent, paragraph, space, vertical skip lines, page length, page numbering, centering, fill, justification, etc. Use with ARY Editor. Supplied with "Structured Source" (Windrush PL/g); easy to see the flow of the program.

F and CCF - \$49.95



•• SHIPPING ••
 Add 2% U.S.A.
 (min. \$2.50)
 Add 5% Surface Foreign
 10% Air Foreign

TOLL FREE
1-800-338-6800

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
 Hixson, TN 37343
 info (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE

*FLEX is a trademark of Technical Systems Consultants
 *OS9 is a trademark of Microware

SPELLB "Computer Dictionary" from Southeast Media -- OVER 120,000 words! Look up a word from within your Editor or Word Processor (with the SPH.CMD Utility which operates in the FLEX UCS). Or check and update the Text after entry; ADD WORDS to the Dictionary, "Flag" questionable words in the Text, "View a word in context" before changing or ignoring, etc. SPELLB first checks a "Common Word Dictionary", then the normal Dictionary, then a "Personal Word List", and finally, any "Special Word List" you may have specified. SPELLB also allows the use of Small Disk Storage systems.

F and CCF - \$129.95

DATA BASE - ACCOUNTING

XOMS from Westchester Applied Business Systems -- Powerful DBMS; M.L. program will work on a single sided 5" disk, yet is F-A-S-T. Supports Relational, Sequential, Hierarchical, and Random Access File Structures; has Virtual Memory capabilities for Giant Data Bases. XOMS Level I provides an "entry level" System for defining a Data Base, entering and changing the Data, and producing Reports. XOMS Level II adds the POWERFUL "GENERATE" facility with an English Language Command Structure for manipulating the Data to create new File Structures, Sort, Select, Calculate, etc. XOMS Level III adds special "Utilities" which provide additional ease in setting up a Data Base, such as copying old data into new Data Structures, changing System Parameters, etc.

XOMS System Manual - \$24.95

XOMS Lvl I - F & CCF - \$129.95

XOMS Lvl II - F & CCF - \$199.95

XOMS Lvl III - F & CCF - \$269.95

ACCOUNTING PACKAGES -- Great Plains Computer Co. and Universal Data Research, Inc. both have Data Base and Business Packages written in TSC X BASIC for FLEX, CoCo FLEX, and UniFLEX.

Call 800-338-6800 for more information

MISCELLANEOUS

TABULA RASA SPREADSHEET from Computer Systems Consultants -- TABULA RASA is similar to DESKTOP/PLAN; provides use of tabular computation schemes used for analysis of business, sales, and economic conditions. Menu-driven; extensive report-generation capabilities. Requires TSC's Extended BASIC.

F and CCF - \$100.00, U - \$200.00

DYNACALC from Computer Systems Center -- Electronic Spread Sheet for the 6809.

F and SPECIAL CCF - \$200.00, U - \$395.00

FULL SCREEN INVENTORY/HRP from Computer Systems Consultants -- Use the Full Screen Inventory System/Materials Requirement Planning for maintaining inventories. Keeps item field file in alphabetical order for easier inquiry. Locate and/or print records matching partial or complete item, description, vendor, or attributes; find backorder or below stock levels. Print-outs in item or vendor order. HRP capability for the maintenance and analysis of Hierarchical assemblies of items in the inventory file. Requires TSC's Extended BASIC.

F and CCF - \$100.00, U - \$150.00

FULL SCREEN MAILING LIST from Computer Systems Consultants -- The Full Screen Mailing List System provides a means of maintaining simple mailing lists. Locate all records matching on partial or complete name, city, state, zip, or attributes for Listings or Labels, etc. Requires TSC's Extended BASIC.

F and CCF - \$100.00, U - \$110.00

DIET-TRAC Forecaster from Southeast Media -- An X BASIC program that plans a diet in terms of either calories and percentage of carbohydrates, proteins and fats (C P G) or grams of Carbohydrate. Protein and Fat food exchanges of each of the six basic food groups (vegetable, bread, meat, skim milk, fruit and fat) for a specific individual. Sex, Age, Height, Present Weight, Frame Size, Activity Level and Basal Metabolic Rate for normal individual are taken into account. Ideal weight and sustaining calories for any weight of the above individual are calculated. Provides number of days and daily calendar after weight goal and calorie plan is determined.

F - \$59.95, U - \$89.95

Availability Legend --
 F = FLEX, CCF = Color Computer FLEX
 D = OS-9, CCO = Color Computer OS-9
 U = UniFLEX
 CDD = Color Computer Disk
 CCT = Color Computer Tape

!!! Please Specify Your Operating System & Disk Size !!!

'68' Micro Journal

Accounting Software available yet, and the Word Processing Software selection is somewhat limited), but there are other important areas that already have good in-depth coverage, such as Data Base Management packages (at least a half dozen), 5 "C" Compilers, an Assembler or so, several Pascal packages, a couple of BASIC's, several Communication packages, numerous "Utility" packages, etc., just to mention a few. As with most "new" Computer Systems, there are a lot of what I call "personal enjoyment" Programs available, such as Hayden's "stable", which allow both Programmers and Users to begin to "get a feel for the Machine". Most Business Packages were waiting on the 512K Mac and Apple's Network, and these also require a lot of "User Interface" re-thinking and re-Programming, but they should begin to appear in the near future.

Since the Mac forces Programmers to "Think like a USER" rather than like a Computer Expert, and because the Macintosh is "Event Driven" rather than "Application Driven", it has turned out to be harder to port other Software over, which has slowed down some of the releases. The Programmers Interface to the Mac is complex and "different", and while it does relieve the Programmer of a lot of the details in interfacing an Application to the System, it requires more time to figure out how it all goes together. This may provide faster Software Development once the Developer learns the system, but it also induces a delay in getting the first products out.

In addition, the Graphics capability of the Macintosh is an integrated part of the System, not a "add-on" as it is on most of the other Personal Computers. For example, the Text that is displayed is not a standard character set generated by a Display Controller/ROM system, but is composed of various fonts which can be manipulated in numerous ways because each and every dot on the Display is controlled by the "System". The Mac is a potent "Drawing" Machine with its Graphics, which is a whole NEW area for using Computers, and may turn out to be as revolutionary as the Screen-oriented Word Processor in the utilization of a Computer. This has thrown a real "Monkey Wrench" into the Development of Software for the Mac, because the Software Development Industry does not have any experience in this area to use as a guideline. Rather than simply developing new methods and procedures in interfacing an Application to the Mac, the Software

Developer must develop a whole new ATTITUDE and APPROACH to effectively use the Mac's Graphics-oriented User Interface. Also, this Graphics capability opens new doors for totally different types of Software, such as Apple's MacPaint and MacDraw, or Telos Software Products' Filevision, to name a couple of examples of products that are different from anything that has been available before for a Computer System. Again, we might note that the copies of the MacPaint package that are showing up on several of the "established" Computer Systems provide an indication of what the overall Industry thinks of this type of Software Package. It will be very interesting to see what types of Software appear in this area as this part of the Industry matures over the years.

Apple appears to be going all out to support the Mac. The common problem of finding out what is in the ROM in the Computer, for example, does not exist with the Macintosh. Anyone can obtain a copy of the "Inside the Macintosh" Manual (Apple recently announced that they have signed an agreement with Wiley to publish this Manual). We have not heard just what form it will be in, but I would anticipate a multi-book set with the possibility of ordering a set of Disks of supplemental information. This Manual will not give you a disassembly of the Mac's ROM (thank goodness!); it does provide information on how the Software System works and how to interface with it in writing your own Programs. For example, you do not have to design the Windows in detail, you simply specify what size to make a Window, where to put it, and what it is to contain, and call "System Routines" that actually draw the Window and control it. Our present 'Draft' Manual is about 6" thick and it's "all MEAT" — each and every sentence must be studied to begin to understand what is in it. Maybe we will see a series of the "Lightning Bolt" Books on disassembling the ROM and System Software sometime in the future, but I kind of doubt it; it sure would be a BIG set of Books. Apple will also be making use of various Bulletin Boards, CompuServe, regular "Newsletters", etc., to help get the information out to those that are interested.

Apple is also supporting the Mac with Hardware; the primary thrust over the next two years will be developing and supporting the AppleTalk Network in a "Business Office" environment. While not as fast as

most of the other "Networks", AppleTalk is SIMPLE and INEXPENSIVE! A \$50 "Y" Adapter for each of up to 32 units allows any unit to be hooked up to the AppleTalk Network; there are no switches to set, etc. Everything is handled in a 5K Software package. For those who need more capability, there will be "smart" boxes such as the File-Server with a 20 or 40 Meg Hard Disk and full 68000-Based Computer System to handle more complex hookups such as communicating with Mainframes, other 32-Unit Networks, etc., as well as perform numerous other supportive tasks (the target release date is summer/fall '85 at around \$5000).

The \$7000 LaserWriter contains a 12MHz 68000 Computer System with 512K of ROM and 1.5 Megs of RAM so that IT can handle most of the "processing" required in generating the Text and Graphics at the 300 dots/in. that it uses, rather than tie up, for example, a Macintosh and special Software, for that job. The LaserWriter ROM does not contain the different fonts in a bit-map format; it contains the outlines of the characters in 13 different font styles. This allows the printed characters to be just about ANY size and have ANY orientation (print at any angle, upside down, whatever), format (such as bold, italic, shadowed, outlined), etc. The ROM also contains a "FORTH-like" programming language called PostScript which allows Software such as MacWrite, or any other piece of Software, to send a PostScript Program, along with ASCII Text, to the Printer rather than the "bit map" that is sent to the ImageWriter (which radically cuts down on the amount of information that has to be transferred to the Printer). This "Program" will tell the LaserWriter what fonts, sizes, formats, etc., to use, and let the LaserWriter's Computer handle the details. The LaserWriter is already set up to use the AppleTalk Network; several Mac's can be hooked up to the LaserWriter and each can use it, with no other parts or pieces needed other than a \$50 Adapter for each unit in the System.

Finally, the overall Macintosh "Operating System", to use the term loosely -- Apple actually considers the Mac ROM and the System Software to be two separate packages, an Operating System that interfaces with the Mac Hardware, and a User Interface Toolbox that interfaces with the Mac Programs -- shows considerable forethought. For example, it already includes provisions for Color; 8 Colors with 32 bit-planes can be written into Application

Software NOW, which will simply show up in black and white at this time, but will be ready when color IS available (rumors hint at a 68020-Based Color Mac in a couple of years). For another example, since everything is soldered into a 4-layer PC Board, how do you "update" ROMs? The System Calls use the 68000 "1010 Trap" (they leave the "1111 Trap" open for the Software Developers' use), so this vector can be diverted to a RAM Routine which can filter out any updated calls if needed; this provides both an easy way to keep the ROMs up-to-date (as long as you keep your "System" Disk updated), as well as allowing Software Developers to use their OWN Routines in place of any of those in the ROM if they so desire.

While there may be some drawbacks (for example, accessing the Disk System is S-L-O-W compared to most other Operating Systems, although Apple expects to have some updates this year that will speed things up some), overall, the Mac Operating System appears to contain a flexibility and expandability that is seldom seen in other Computers, and only time will tell how efficiently Software and Hardware Developers can learn to use the System.

A "Macintosh System" comes in many flavors. The basic 128K, Single Disk Mac is definitely a "Starter" system; it DOES get you a Mac with basic Word Processing and Graphics capabilities, so you can begin to get familiar with the System. But to have a "Usable" system, you really need the external Disk Drive (there is only enough room on the normal "System" Disk for one or two Data Files -- the Mac, especially the 128K Mac, uses a lot of Disk space for "virtual storage") and a Printer. The "System" Printer is the Apple ImageWriter, which is basically a "Pro Writer" with Apple Electronics, but there are Drivers available from external sources which drive Epsoms, Daisy Wheels, etc. Our experience indicates that a "minimum" System would be a 128K Mac with the external Disk Drive and a Printer if you expect to use the System for much of anything. Double-Sided Drives, which the "rumor mill" says should be available this summer or fall, will be a BIG help.

While other Printer Drivers for other Dot Matrix Printers are becoming available, they are not the "Complete Answer". I would guess that the ProWriter was chosen to be the System Printer because its dot pattern of around 80 dots/in most nearly matches the Mac's Display Screen, providing a Printout that almost exactly matches the

Display in size and texture. We have been using an Epson FX-80 while our ImageWriter is "in the Shop" (more a little later), and have found that its 120 dots/in produces a much 'cleaner' Printout of Graphics Displays, etc., but the SIZE is different. To get a clean "High" quality print, the Printout is wider than the Display. This becomes very noticeable with different Fonts, etc. While this situation may not be a problem for you, you should be aware of it.

The reason our ImageWriter is in the Shop (now for the third or fourth WEEK) is that one of the Pins in the Print Head froze up. So what's the problem? Well, first, it costs almost \$100, and is not easy to change (the Epson Print Head costs \$9.95, and can be changed in a few seconds). Second, they are SCARCE; while "no one" knows about any "unusual problem" with these Heads, even the MAJOR ProWriter Service Centers don't have them. The local Apple Repair Shop assured us that they could have it repaired in a couple of days, and after a few days of "working" on it, decided that the Printer needed a new Board ("updated" Board, as they called it). Well, OK, we need to have the most "up-to-date" Board in our Printer, right? After the Board was replaced, we were informed that the Print Head was bad --- SURPRISE!!! So, we have been using the Epson the past several weeks while waiting on a 2-day repair. And we are not alone in this problem (re some of the "Letters to the Editor" in other publications). Hopefully, they (??? whoever) will develop a decent Print Head in the near future.

Is the Mac for you? Only YOU can determine that. The Macintosh DOES offer many options and possibilities that we have not had in the SS-50 Bus community; most notably that there will be a LOT of them in the Marketplace, and have a LOT of different Software and Hardware to use with it. It also offers a lot of options and possibilities that have not been available before in the Computer Industry in general, such as the flexible but simple Window/Icon User Interface, the Graphics capability, and the AppleTalk Network (an IBM PC Board is being developed to interface with the AppleTalk Network, while the FileServer Unit will provide access to "Mainframe" Networks). Whether it will develop into a force in the Business Computer arena, only time will tell, but it seems that Apple DOES have a "viable alternative", and Marketing Force to capture a large chunk of the Personal Computer Market.

Ramblings

Mr. Don Williams
68 MICRO JOURNAL
P.O. Box 849, 5900 Cassandra Smith
Mixon, Tennessee 37343

Dear Don,

I was a bit startled by Ron Anderson's comment in the March '85 issue (Page 10, column 1, Paragraph 4) that TSC seems to be getting out of the 68xx business. But I looked for their advertisement in some of the months preceding that issue and, as sure as an indicator could be, no advertisement. Horror of horrors! Now not only am I obsolete in equipment but my DOS has gone and gone obsolete on me too.

In fact everything related to computers I have is obsolete. I built my first SWTPC 6800 system about 1978 and that went obsolete when the 6809 came out. Yet that was working fine when I rushed off my money for my second SWTPC S/09 system in January 1980. In fact that original 6800 system is STILL WORKING FINE TODAY as I sit here and write this letter. True, I haven't used it for some time. But why couldn't I use it as a parallel processor sometime? Simply hook it up through a parallel port to the 6809 S/09 and boy could I have some operation run like greased lightning.

Talking about obsolete, I wanted to try some graphics but neither SWTPC system had any ("We Don't Play Games"). I looked at the boards available and they were either too expensive or too crude. Why not buy a cheap "appliance" computer and connect it to the S/09 through a serial or parallel port? Well you never saw a computer go obsolete faster than the Texas Instrument TI-99/4A. I would have reported on this toy through your Journal (the graphics were good at the time for the money involved), but I got mad at them when they charged more for the box to interface to the serial port than the computer cost! I did not want anyone to buy one based on my recommendation after that.

On my third version of FLEX9 I vowed to heaven I would never, ever, buy another version of FLEX. Sure enough, UNIFLEX came along. But nobody ever convinced me why I should pay several hundred American bucks for a multi-user system when I am only a single user system.

Recently I had an occasion to use a brand new Radio Shack Model 4P. In my opinion, that computer has not yet matched the capability of my SWTPC S/09. True, I didn't get a routine to tell the time with the S/09. I have thought of placing one of those stick-up digital clocks on the terminal. But if I turn my head slightly to the left I can look at a regular clock to tell the time. If the computer wants to tell the time he can ask me nicely and I may write a program for him to use his two on board timers. But the Radio Shack did have one very superior advantage. The documentation was beautiful. If I had that documentation when I started computing, I would have saved all that time reading one piece of inscrutable prose after another. But I would have missed all the fun too.

But there's the bits too. My bits are going obsolete. The 6809 is only 8 bits. There's 16 bits and 32 bits coming. What am I ever going to do with all those bits? "Ah, but you can calculate ever so much faster!" you say. "Why can't I use my 8 bits and calculate with my on-board calculator?" I answer. "Surely the calculator chip will calculate much faster than the 16 bits or the 32 bits." "Not if we give them more mhz." "You mean my mhz is obsolete too? Oh! that really hurts!"

Now the Fat Mac at 512k does make my mere 128k obsolete. Everybody seems to be in the big memory business these days. I seem to recall that in the early days of micro-computers a sieve-of-Erastosthenes required less than 1k. Now, according to Mr. Edward Joyce who wrote in the October 1984 issue of MICROCOMPUTING (not another obsolete!) magazine, page 63, Logitech's Modula 2 REQUIRES 18,970 bytes of object code (NOT INCLUDING AN 8kb RUN TIME PACKAGE). I have an idea, why not keep all our programs in RAM. Then we wouldn't need disk drives any more. (LOTUS 1-2-3 anyone?)

Some time ago, perhaps in the '60's, I read a great article in ANALOG, the Science Fiction magazine. Perhaps someone will write in with the date of the original magazine, for that article is really worth reading. I do not remember who wrote it or the title (but it may have been "The All American 5") but I will try to recall the high points here.

Back in the '30's the electronics industry had achieved a 5 tube design for a radio. It was called the "All American 5". That's all it took to make a radio. Just the 5 tubes. No more. No less. Some clever manufacturer figured that if he put more tubes in the radio he could sell more radios. And the big tube race went on. There were 6 tube radios. Then 10 tube radios. Even 20 tube radios. Some manufacturers even had the extra tubes light up so you could look in the back and see all that great work being done.

Then it was cylinders. The more cylinders a car had the better it had to be, didn't it? The cylinder race was on! 6 Cylinders. 8 Cylinders. No. Not 10 cylinders? Yes and even 20 tiny little cylinders.

I guess you are all too young to remember the watch jewel race. At first everyone thought that there were really diamonds in those watches. They just had to be worth more if they had more jewels. The jewel numbers continued to climb until some sourpuss squealed that the jeweler's term for a normal bearing was "jewel" and that was all over.

Back to the radio again with the transistor. It took 7 transistors to make a radio. Superheterodyne and all. The transistors got at least to 19 as I can recall from my personal memory. Nobody would sell a radio with less than 10 transistors.

Wattage anyone? I can get thrown out of my apartment if I play my 3 watt Webcor (sad! another obsolete!) stereo record player too loud. The lowest wattage stereo receiver I could find was 15 watts. How does it happen that people buy 100 or 150 watt stereos for their houses? I believe that on the day of the Last Judgement only God himself will use 150 watts to announce the end of the world and the end of the wattage race I hope.

So the races now are with bits and bytes and mhz and ram instead of with tubes and cylinders and jewels and transistors and watts. My computers work very fine. I fear the day when I can no longer get 8 inch diskettes (Not Not 5 1/4 inch! Not 3 inch! Arrgggh!) or print ribbons or stuff like that. When I can no longer get the raw material to feed to my computer, only then will it be truly obsolete.

But isn't the number of people who own obsolete computers growing every day? Bally, Intertec, Adam, Altair, MITS., Osborne, Sinclair and on and on. Isn't the IBM PC shortly to be included on that list because of the PC AT? Shouldn't there be a magazine called the OBSOLETE COMPUTER? Wouldn't that magazine have a marvelous circulation?

Yours truly,


Clifford Glennon
3395 Nostrand Ave. Apt 2G
Brooklyn, New York 11229

Ed's Note: Clifford, you just hit the nail right on the head. A lot of brands and models have become obsolete. They were planned, designed, manufactured and sold, all with obsolescence built in. That is the American way to merchandise a 'successful line'. It is the foundation of almost all major manufacturers, otherwise, they would soon run out of repeat customers. The auto industry is just one example; they got 'eaten alive' by the Japanese. Because sometimes it became the same - junk and obsoletable (think I just invented a new word). Fact is I have been told by designers that it was difficult to design in such a way as to insure it would fall apart in a predetermined time. At times falling apart was not the design goal, just let it fade away, without adequate support - forget it and on and up to the new improved model. Bye, Suckers! Remind you of anyone?

I own both a late model Caddy and a Chevrolet van, in addition to a 1972 Fiat convertible. I hesitate to call it a 'sports cars' lest any of you get the wrong idea. Also I have a '69 Ford pickup with over 100 grand showing, and a lot not showing. So let me tell you about my experiences with obsolete.

Notice I have not included obsolete computers in my ramblings so far. I own a couple of those also. More later on them.

As a result of the Cadillac having so many problems and faults, as well as the Chevy van rusting away due to poor prime and paint application. Not to mention the engine that drinks oil like it was going out of style. And the dash that sounds like a Texas rattler, even on paved road. The whole thing is a rolling pile of obsolescence. When I go into a Chevy dealer, they just attempt to shrug away my complaints, and start telling me what a great job the newer one will do. Well, I will never buy another Chevy or GM anything, ever, I am now on public record of getting a case of the smarts. However, I guess a

little of it is due to the fact that I know somewhat more about computers than I do automobiles. Also, for as long as I live and buy autos, the experience with Ford will make me a Ford customer. It has kept running and gave me more than I bargained for. Also the Fiat has run like a top, and everything still works. They dropped their USA distribution, but I can still get parts. I have a hassle trying to get parts for my van, and it is a '79 (parts that is that have some quality).

Now as to Obsolete Computers.

I was talking to a friend and reader who is employed by the FAA the other day, he was telling me about one of the S50 bus computers they were using in some of their operations, at a regional office. He was buying additional hardware, from Data-Comp for the system. I asked him what it was, I was told it is a 7 year old SWTPC 6800 computer and ran at least 8 to 10 hours daily, seven days a week. And they can still get any part for it they need! I thought about this, especially since getting your letter Clifford. Today I called a reader and user, as well as Data-Comp customer, who is a NASA engineer, and he tells me they have several (meaning 'many') GIMIX and SWTPC systems that are at least 5 to 8 years old. Fact is a CoCo running S.E. Media FLEX modified by S.E. Media was the computer that launched the FIRST commercial, non-government USA space vehicle. Those systems are still running, the software is not in the least obsolete. The CoCo, as we now know it, may soon be dropped by Tandy (remember the TRS Model 1, model 12 and soon model 4, etc) but the aftermarket will probably keep it alive, due to the large numbers of them sold.

I have not heard a word from Wavemate for a couple of years, yet S.E. Media sells software all the time to users running that machine. And we can still get practically any part needed also. Then there was MSI, gone for some three or four years and I know many still running and being supported, because of the similarity of S50 bus systems. And even as I type this, I received a call from a user of SSB equipment, 6 years old, still running the DOS69 and completely satisfied with the system. And any of these systems can run the 68000 also. Obsolete?

FLEX is not, in my opinion, supported by TSC any longer. In fact the only thing that I know of that they have done with FLEX in

the past few years is jack up the price. But that is another story. They still support UnifLEX, and in fact a very nice version (virtual memory) will soon appear on a new 68020 machine from GIMIX. However, I must admit that most users of FLEX and UnifLEX, I talk to, do not feel that TSC has kept the faith. Yet, I do keep up, in various ways, with what is occurring and it is my personal opinion that their decisions in respect to FLEX especially and the S50 in general was a mistake. I will be glad to retract this given sufficient information to the contrary. UnifLEX users have had it somewhat better, but not to the extend previously expressed or implied.

That any manufacturer of either software or hardware should restrict his offering to the S50 bus is patently unthinkable. All of us would desire that we have a profitable growth and expanse of our market. We all felt good when some of our suppliers made it on the 'other side', but not when they did by ignoring us who had given them their start. When I went higher in grades (schooling) then my parents did, I did not forget them and go out and search for a new Mom or Dad! Nope, I just rejoiced and thanked them for making it all possible. That kind of thinking goes equally as well in business.

I am thankful to fine folks like those at SWTPC, GIMIX, SSB who were there at the beginning, and never made their stuff obsolete. I can still get practically anything I need for any of those basic systems. They have improved the breed a thousand fold, but in the process never made my original system obsolete. Without that we would have been gone long ago.

For the past 5 or 6 years I have watched OS-9 grow. Both in quality as well as quantity. OS-9 is now used on more different brands of 68XX(X) computers than any other disk system - period! Yet, despite its growth, and expansion to other systems and CPUs. Microware has not seemed to have forgotten their beginnings.

I have not always agreed with some or their policies (as with many others), and matter of fact many of them have not agreed with ours. Yet, they have not obsoleted any part of their product. I thank them as well as all the other software and hardware vendors and manufacturers who still remember who we are.

Now I will tell you a documented truth; those who have kept faith with their beginnings and loyal base of users, and who offered a reasonable quality product have

survived. Not many got rich, but most eat fairly well. And that IS important! They will still be around when it is feasible to make that expansion, but not if they forget their base of users. I cannot tell you how many have lost valuable credibility because of the 'bad-mouthing' they received from their users because of their apparent lack of support of previously sold products. And I personally know of some mighty fine contracts that were made possible because of a proven track record of GOOD support. I don't care how well you might think your product or support is; if your users feel you fail - **YOU FAIL!** The IBM PC (tm) is not the best micro - but look at what reputation has done out in the marketplace.

So Clifford, I guess we will always have shoddy products. Those will be culled out in the normal course of business progression. For those who remember, have a good chance at survival, because we are loyal. But the sad part is that we will lose, and I mean WE LOSE, when any one of our basic suppliers becomes a little 'big headed' and forgets who gave him his start.

If ever you think that we are letting you down, I want to personally know. I know who made us and I thank each and everyone one of you, reader and advertiser alike, from the very depths of our hearts. **WE REALLY APPRECIATE YOU!!!!!!!!!!!!!!!!!!!!!!**

DMW

Computer Publishing Center
68-Micro Journal
5900 Cassandra Smith
P.O.Box 849

HIXSON. TN 37343
U.S.A.

Raymond Canneuf
Berliner Str.34
6437 Maintel 1
W. Germany
Europe

Maintel. den 19.2.85.

Subj.: Upgrade the old black box

Dear Editor,

The letter from Mr. Piacenza published under 'Bit Bucket' in the december 1984 issue expresses one of the dreams of every computer freak I guess. A decent hardware, a flexible system layout and a high reliability in order to run multiple operating systems, without board swapping or chip-exchanging for going from one OS to the other, and all of this for a reasonable price! However, being aware that nobody will invent an egg-laying wool-milk-pig, I might introduce my setup as an example how an older system can be upgraded with (minor) modifications in order to realize the above mentioned dreams. I do not claim that this is the ultimate system, but this contribution could show to the 68Kw-fraternity here a way for an upgrade, using most of the already available hardware. For the time being I can switch-select between FLEX, UNIFLEX, OS-9 and CP/M using the same hardware configuration.

In the early days, as South West Technical Products still manufactured kits, together with some other fellows here I started out building a 6800-klt from the scratch. So after a while I also was a proud owner of such a black box.

More or less at the same time some rumours came up that the 6800 would be replaced by the 6809. Also around that time Southwest moved from the 'hobby-market' into the 'system-business'. Prices went up for the last bare boards, new software was made only for the 6809, etc. Many self-dealers at this side of the atlantic tried to get rid of the hobbiest. Help was no longer available.... They smelled 'business' and knew only the words 'bookings' and 'profit'. So dark days began for the 6800-owners.

At this time I decided to convert to 6809. But prerequisite however was to use as many parts as possible from my old system. First step was to convert the old MP-82 motherboard to the 6850c-bus-lavout. Next step was fully decoding of the old mother-board. Later I wanted to install the virtual disk, so extended addressing was required. Therefore I had to modify some of the available memory boards. After a Stateside visit I came back home with an older version of OS9. So the next project was to be started. In order to get the OS9-clock running there was a need for timer on address 4E09D. For some reasons I had to convert the I/O-ports from 4 to 16 addresses per Port. The next step was to make a task manager in cpm, hoping to open my system for the so called over-whelming world of application programs, which are told to be available for the cpm-community. Finally, some time ago I installed Unix on the same msinframe. Some other modifications were required to make it running, but still having access to the all previous features.

All of the above mentioned items are described in the next chapters.

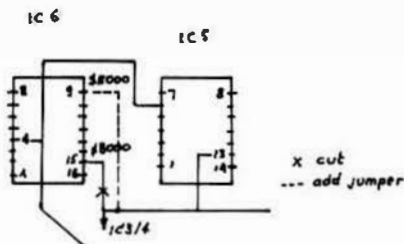
1. Conversion HP-B2 motherboard for 6809 cpu (HP-09A board).

Several modification notes were published in the 68MJ in the past, so I will not repeat all of the thoughts which have been written on this subject meanwhile.

```

First of all the I/O-ports must be moved from the $B000-area
into the $E000-area.
the supplied MP-82-manual, page 4, describes how to do this:
e/ cut path between IC6/15 and IC3/4.
b/ connect IC6/9 to IC3/4.
  this moves the I/O-block to $E000.
c/ cut MRST from MP-82.
d/ cut U01 & U02.
e/ connect reset switch with 2 wires to the MP-DP board.
f/ it is useful to connect an abort-switch to the nmi
  controler of the CPU-to-IO.
  At this moment I could boot flex9, using my old
  DLX-controller for the M808.

```



2. Address-decoding.

Having purchased a graphic board, addressed at 0E100, I ran into an addressing problem. So I decided to to decode the 0Exxx-area for my needs.

A good help for this job was an article in the MJ68, issue April 82, page 35.
Since I also replaced the DC2-controller by the DC4 meanwhile, which allows 16 addresses per port, I decided also to modify the port decoding from 4 to 16 addresses.

a/ cut addressing A5 to Ic6/4 on the HP-82 mother board.

b/ using an 8N7425 and an 8N7400 make following decoding:

```
SN7425 pin 1 to ground
          2 to A11
          4 to A10
          5 to A9
          9 to A8
         10 to A7
         12 to ground
         13 to ground
          6 to SN7400/1
          8 to SN7400/2
```

8N7400 pin 1 to 8N7425/6
2 to 8N7425/8
3 to 1c6/4

After this modification the I/O-decoding is 0E000, 0E010, up to 0E070. So the serial card for the crt must be moved from slot 1 to slot 0, and the floppy controller must be installed on slot 1 (0E010). Do not forget to remove the wire from slot 5.

These chips can be positioned upside down at the left side on the components side of the KP-82 motherboard by use of some adhesive. Now I could plug my graphic controller on to the 5550-bus, which allows us now to create 512 x 512 pixel pictures. The video monitor used is the old C164 from southwest.

3. More address decoding.

The flex-utility 'TIME' requires a timer at address \$E090. Also for my Q59-implementation there was a need for a timer at this address. However neither the timer, nor the address decoding for this address was prepared so far. So lets do some more decoding:

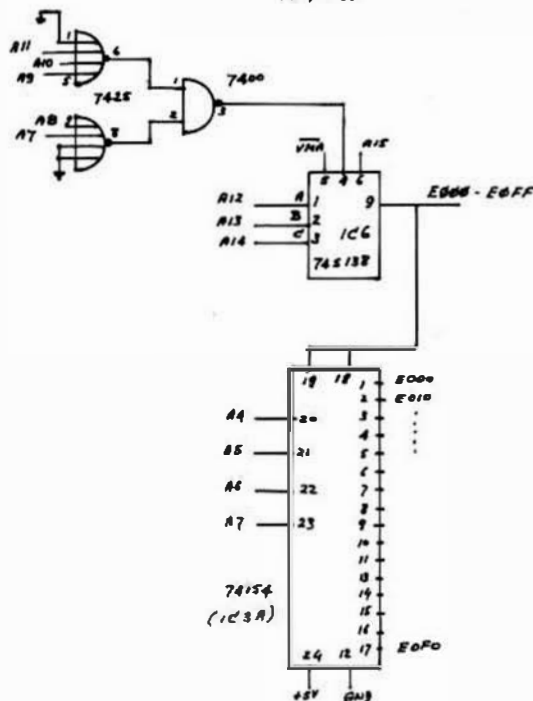
```
#/  first step is to replace the original 74S138 (ic3) on the
MP-82 motherboard by an SN74154 decoder.
Therefore I built a small adaptorboard, that plugs into the
original ic3-socket on the MP-82 motherboard. the 24pin
socket for the 74154 is hardwired to a 6pin socket on the
same adapterboard. The pins of this auxiliary socket plug
into the old socket on the motherboard.
```

b/ following connections were made:

748138	74154	address	port
1	23		
2	22		
3	21		
4	18, 19		
5			
6			
7	8	E070	7
8	12		
9	6	E060	6
10	5	E050	5
11	5	E040	4
12	4	E030	3
13	3	E020	2
14	2	E010	1
15	1	E000	0
16	24		
A7	20		
	9	E080	8
	10	E090	9
	13	E0A0	10
	14	E0B0	11
	15	E0C0	12
	15	E0D0	13
	16	E0E0	14
	17	E0F0	15

In this way I got all the required ports, also some extra ones for later expansion.

Page 5a decoding 16 addresses per part
16 parts



4. Timer on 0E090.

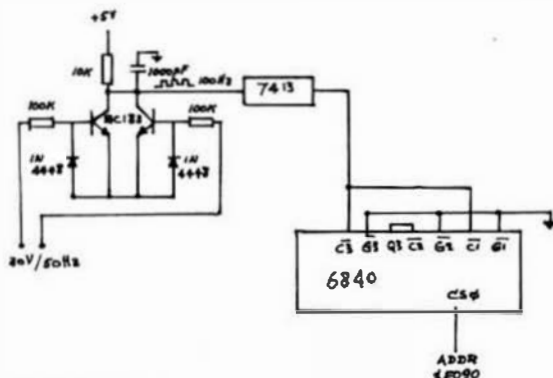
the newer swtp-computers have a total new design of the motherboard, including a MP-ID card. This MP-ID card hold the 6840-timer.

the design. I would build this timer on a 8930-prototype board. This board will also be used to hold the baudrate generator, which will be covered later.

I copied the timer as published in the MF-10-schematic. After completion of the circuit, I found out that it would have been easier to use a bridge rectifier instead of the two transistors. But I was too lazy to modify the circuit up to now. Everything works okay so far. ...

The 50 hz line frequency is doubled by the transistor circuit, and is sent to the 6840-timer through a 7413 buffer/trigger. (see schematic)

This gave me the required timer for both 'TIME'-utility, and 059-adaption, which will be covered later.



5. Extended addressing.

In order to run the 'UDISK'-package, extended addressing must be enabled, both on the cpu-board and on the 5550-bus. Therefore two modifications are required:

- a/ install the DAT-RAH on the cpu-board

The cpu-board may have either ic8 (DAT) or ic24 (timer) installed. (NOT BOTH!) For extended addressing ic8 (DAT) is required, so ic24 must be removed! see next point. Referring to some notes I found, ic8 745189 should be an 'S'-type. Be careful while installing the chip or the socket. Do not bent any pins and take care to make good soldering work! A lot of troubleshooting can be avoided in this way.

In addition to ic8, resistors R21-R24 (470 ohm) must be installed.

See also application note 122a from atp.

Referring to the MP-10-schematic, and an application note from southwest, I also removed R4, R5, R10 and R6 from the old MP-B2 motherboard. R9 is replaced by a 600 ohm resistor, and FIRE (UD2) connected to +5V through a pullup resistor.

At this point I have to say thanks to SMSCHV, my friend Hans in Uppsala. He is one of the sub-representatives who takes care of after-sale service!!

- b/ remove the baudrate generator 14411 (ic24).

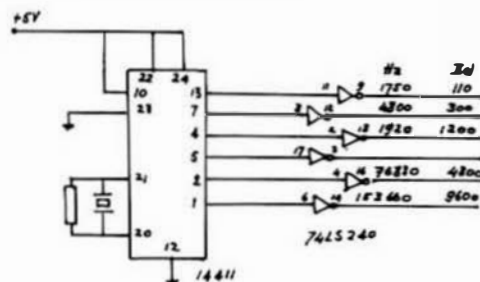
this timer can be installed on the same prototyping board, which is used for the 8E090-timer. (see chapter 4)

- c/ cut the baudrate lines 50-49-48-47-46. Those paths cannot longer be used for timing purposes. The 5550c-bus uses those lines for extended addressing A16-A19. So cut the paths between the 5550 and 8830 section.

- d/ install the 14411 on the prototyping board.

the same circuit as used on the cpu-board can be build on the prototyping board. (see schematic)

the appropriate clocksignals must be sent to the baudrate lines on the 5550-bus section.



6. Coding the memory boards.

Now everything is setup to make use of the extended address capability. I use two static 64-K-ram cards from Digital Research. I had some problems in finding the correct switchsettings on the board. Finally I found following coding to be used with the boards:

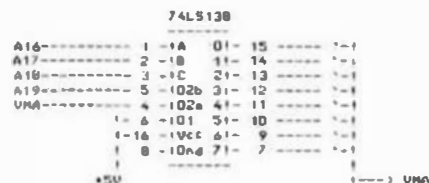
--> on	S0	S1	S2	S3	
1 1 1 S3	1	1	1	1	board 0
1 1 1 S2	1	1	1	0	board 1
1 1 1 S1	1	1	0	1	board 2
1 1 1 S0	0	0	0	0	board F
1 1 1 ext.addr. enabled					
1 1 1 C	98000	-	9BFFF		
1 1 1 B	84000	-	87FFF		
1 1 1 A	60000	-	63FFF		

After successful coding and installation, the prompt '50U01.x 56K' will appear on powering up the system. After booting the system, the flex-prompt FLEX 2.MK 102 k will appear.

A little bit of experimenting with the boards showed that board 0 must have 8C000-8DFFF enabled. 9E000-9FFFF must be disabled. Boards 1 to 3 must have the range 8C000-8DFFF disabled.

7. More memory.

Since I also had an old 32K dynamic ram from Digital Research available, I also wanted to plug in on to the 5550-bus also. First step was to prepare this board for extended addressing. A useful article was published in HJ68, issue sep.83, page 28.



With the switch arrangement one can select on which page the 32K-block will be located. This small additional circuit can be positioned on the lower right corner of the board components side.

A nice program to make the used memory blocks visible is the 'MEMMAP'.

With this installation so far, and running 'DYNASHAR' I can have my own running his basic-program in one block, while having running the assembler or something else in another block, both using an individual cr1.

this is a nice alternative to the real multi-user systems.

8. 059

I have always been interested in 059, but I also always want to be able to turn back to Flex without swapping boards or replacing chips.

A minor modification on the MP-09A board allows access to both operating systems at a flip of a switch.

The 059-firmware exists of two 2716-eproms. The first eprom should be placed at address 9F800, i.e. at the same location where the 58U0 is positioned. The second eprom can be placed at any location of the free eprom sockets. Since I have scratchpath at 8E000 (ic2), I decided to use IC1-socket for eprom P1. Therefore the addressdecoding from 8E000 must be modified in 9F800. The second eprom is located in socket IC3 (9F000).

The modification is simple. It is more difficult to describe however. So I guess the schematic will clarify all questions.

- a/ cut trace between IC6/13 and IC4/20. This should be done on the components side, between crystal and IC16.

- b/ cut trace 81/8 to IC1/20. This should be done on the backside, just above IC1/24.

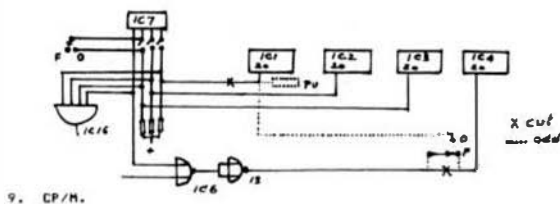
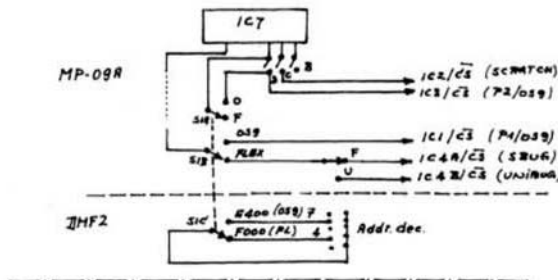
- c/ connect a pullup resistor between pins 20 of sockets IC1 and IC4 to 5 volts

The next two schematics will make everything clear. My 059-version was configured for a DMF2 located at address 9E400, while flex used 9F000. So I decided to use one sector of the 4-position switch. I wired everything to a switch on the frontpanel, so I can flip easily between flex and 059 now.

After successful completion of this job, the 059-welcome prompt will appear on the screen. I am running flex and 059 in this configuration for about one year meanwhile.

Since I only could use the DMF2-controller. Unfortunately Microware doesn't support the DC4-controller I am using for my minifloppies. Someone here promised to help me for this application, but after about 4 months it turned out to be just a bluff. At this point I have to say thanks to the MJ68-team and especially to Joseph Aulicino in New York, who gave me a telephone call almost in the middle of the night. Joe provided me with the required startup support for the DC4-controller. So now I can use either 8" or 5" drives on OSV.

Thanks Joe, this was great!!



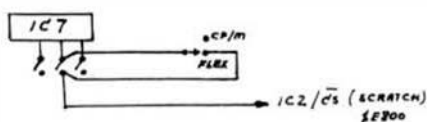
Next application on the same mainframe is the installation of CP/M, using the METALAB-softboard.

The only required modification at this point is to disable the scratchpath at \$E800, which I use for 'flex'. This can easily be done by installing an additional switch according to the next schematic.

Now, after pushing the 'reset'-button, and hitting the 'D' you get the 'Metalab'-prompt. Now CP/M is up and you have access to a wide variety of software. An advantage of the operating system, which we are still missing for the SSSD-bus.

Unfortunately for the time being I am limited to 8" only. Metalab apparently doesn't support the DC4-controller. I never got a reply from Metalab neither on my request for the DC4, nor on the double density. Maybe this company doesn't exist anymore??

If anyone out here in 48Kx-land has either a DC4 or a double density double sided application (or both) up and running, I would appreciate to share some info.



10. UNIFLEX.

Prerequisite to get UNIFLEX up and running is: UNIBUS bootprom at least 128k-RAM 8" drives a serial port at address \$E000 for the crt a crt with 'lower case' capability (a HP-D9B cpu-board), however I am still running with my HP-09A board without trouble so far.

On my system I had to perform two modifications: 1/ modification to select either SBU6 or UNIBUS this is done on the cpu board. 2/ modification of the DPS-serial I/O-board.

1/ CPU-board.

Since all sockets on the HP-09A board are in use meanwhile, I came up with piggybacking SBU6 and UNIBUS. Both eproms are carefully soldered together, pin by pin. With exception of the chip-select line (pin 20). In addition to this, I also soldered a pullup resistor between pins 20 and 24 on both eproms. Pins 20 of both eproms are connected to a switch, which returns the selected eprom to the cpu-board, socket IC4, pin 20.

2/ DPS-board.

I am using the GP5-board from AAA-Chicago on port 0, so with the actual installation I have \$E000 on one acis (U4), used for my serial printer, and \$E004 on the other acis (U3) used for my terminal.

It is more difficult to describe the required pin on the board than explaining this by a schematic. All what we have to do is to cut the two paths going to the CS1-pins of both acis. So cut the trace going from U2/11 to U4/10, and from U2/14 to U3/10. This can easily be done on the components' side of the board. Wiring as required according to the schematic is done towards the 'TERM-A' socket, which is not used in my setup. In this way I can always disconnect the wiring, and pull out the board if necessary.

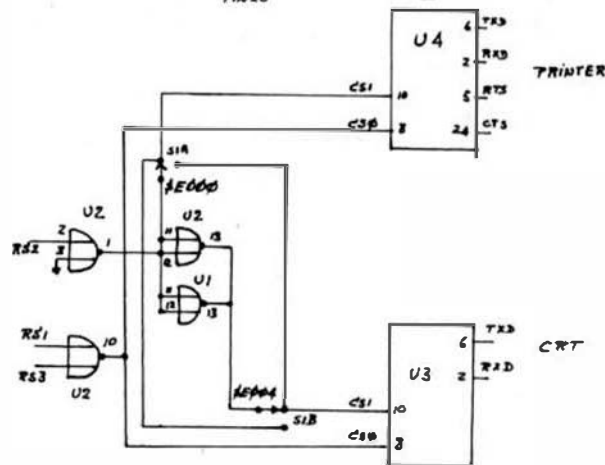
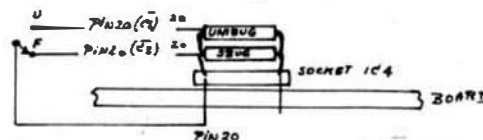
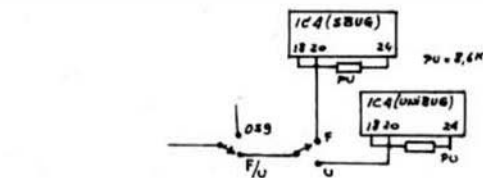
A couple of 12V-relays and a spdt-switch out of the junkbox let me select either \$E000 or \$E004 with the required eprom, so I can easily flip between 'flex' and 'uniflex'.

This application is up and running for a couple of weeks meanwhile without having seen any problem so far.

The jumper installation on the DMF2-controller seems to be rather critical. On my board I made the following jumpersetting:

halt/bus req	bus req
upper/lower	lower
ba/ba & bs	bs & bs
irq/nora	irq
2mhz/1mhz	1 mhz
precomp/nora	precomp
nr/nora	nora
56/32k	56k
desel/ssl	ssl

As mentioned before this application runs only for a couple of months meanwhile without trouble. Having only a few experience with uniflex so far, I can neither recommend nor warn running uniflex that way. I would very much appreciate getting some experience (if any) from other fellow-hobbyists.



DPS Address switching \$E000
for uniflex
board located at slot 0 (\$E004)

Conclusion.

This certainly is not the ultimate system. But I hope I could give some inputs how an older box can be upgraded step by step, as money and time and parts are available. Running the old black box this way expanded my capabilities, and gave me the opportunity to run software from a wide variety of sources.

Sincerely,

R.C.

Raymond Casneuf

BIT BUCKET '68' MICRO JOURNAL

Memo: New Dimensions of Articles, Letters and All Other Items Submitted for Publication on Paper.

To All Subscribers, Readers, Authors and Advertisers,

In an effort to improve the quality of 68' Micro Journal, we are requesting from all a change in the width of articles submitted on paper for publication. Previously 68' Micro Journal requested articles not submitted on disk, to be 7" wide. In order to improve efficiency in reducing articles while maintaining readable quality, 68' Micro Journal is now requesting that all articles (not submitted on disk) be submitted 4 1/2" in width. This change will include all items submitted on paper (including Bit Bucket and Product Announcements, Releases and letters to Ed.).

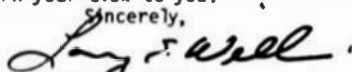
Source listings can be difficult in this new format. While a single line of code can be less than 4 1/2" in width, the longest line in a source listing should not exceed 4 1/2". For this reason 68' Micro Journal is requesting compliance if at all possible, but if not possible, then we are requesting that the longest line in a source listing be made as close as possible to the new width standard. The new changes will benefit both 68' Micro Journal and its readers. All items and articles submitted on disk or paper should follow the new guide-lines as published on page 2 of 68' Micro Journal effective April 1985.

Please try to understand that this request is an honest effort to improve the readability of future publications. 68' Micro Journal will not turn away any article or other items because of non-compliance, but their quality would be greatly enhanced under the new width requirements.

As always, 68' Micro Journal is requesting that all submitted materials on paper be done in dark ink. Please use a dark dark ribbon !! Your consideration in this matter is appreciated.

68' Micro Journal would prefer all items be submitted on disk, following the instructions stated on page 2. We understand that this is not always possible. We will always return your disk to you.

Sincerely,


Larry E. Williams

68 Micro Journal
5900 Cassandra Smith
Hixson, TN 37343

William Lee
8778 Bullock Dr
San Diego, CA 92114

Dear Don,

Here are two utilities the OS9 users out there may find useful. The first, "PRINT", provides for some formatting of text files as they are listed. The other, "NAME", allows the changing of disk volume names.

PRINT:

The format for calling PRINT is:

```
print filespec [title] [>redirection]
```

'filespec' is the pathlet to the text file to be listed and 'title' is an optional header string to appear at the beginning of each output page (40 characters max). PRINT outputs to standard output, so redirection can be employed to send it to a printer.

PRINT divides the file into pages as it outputs them. Each page begins with a header line which contains the title (if provided), date, time and page number, beginning with 1. The header is followed by four blank lines, then 55 lines of text.

PRINT begins by setting up a header line image in memory. It then processes the command line, opening the text file and adding any provided title. The main loop at 'ru_file' calls 'newspace' to start a page, then loops in 'ru_line' until end of file, or 'page.length' lines have been output.

'820' converts page numbers to ascii decimal. 'kill8e' removes leading 8s from the page number. 'eetdate' inserts current system date and time into the header. 'String' outputs the page header. See the listing for further information.

NAME:

Format for NAME is:

```
NAME drive_spec new_name
```

'drive_spec' is the device name, such as "/d8". 'new_name' is the new volume name for the disk in the specified device.

NAME works by opening the specified drive in the mode that treats the entire disk as a file. It reads logical sector 0, inserts the new name with the last character's high-bit set, and writes the sector back out.

Both programs are written in DGS assembler. The pseudop 'var' works like 'rmb' in the Microvare assembler. 'com' is a 'rmb' usable by the external library subroutines, and can be replaced with a 'rmb' if all routines are merged into a single file.

OS9 Paged-Print Utility

```
% program is a formatted list utility
% Takes its input from path provided on
% command line, sends result to standard
% output path. Performs pagination, page
% numbering and naming.
```

```
% First entry: 30Mar84
% 31Mar84
```

```
nam Print
```

```
ifp1
type Formatted List Utility
nif
```

% constants:

```
ff equ %c form feed
eof equ %d3 end-of-file
stdout equ 1 output path
stderr equ 2 error path
isopen equ %84 open path
isrdln equ %8b read line
iswrln equ %8c write line
fsperr equ %0f print error
fsexit equ %06 quit
page.length equ 55
```

% variable requirements:

```
argv 0
```

```
% (for the top-of-page information)
% header appearance:
% count,%c,%d,%a,%a
% title chars
% %20,%20
% yy/mm/dd hh:mm:ss
% %20,%20
% PAGE
% ppp
% %d,%a,%a,%a
```

```

header var 5 form-feed + blank lines + #
title var 40 page heading chars
var 2 for separation
date var 8 date
spaces1 var 1 separator
time var 8 time
spaces2 var 2 before page #
word_page var 5 'PAGE '
page var 3
ender var 4 blank line

print_size equ .-1 (disregard count)

% other requirements:

number comn 5 for binary - decimal
pageno var 2 current page #
lines var 1 current lines on page (0-56)
ipath var 1 input path #
parmsx var 2 pointer to start commline parms
buffer var 128 input buffer

% begin actual code:

start
    stx parmsx save pointer to parms

% preset page #
    ldd #0 binary part
    std pageno

% preset date/time area
    leax date,u
    ldd #"/:
    sta date+2
    sta date+5
    stb time+2
    stb time+5

% preset blank lines
    ldd #sd0a
    std header+2
    std ender
    tfr b,a make two LF
    std ender+2
    stb header+4
    ldd #print_size<8+ff
    std header

% preset the title area to spaces
    leax title,u
    ldd #s20<8+42 42 % space

space.put
    sta ,x+
    decb
    bne space.put

% put spaces between date-page
    ldb #s20
    std spaces2
    stb spaces1 single one there

% preset word 'PAGE '
    lda #'E
    std word_page+3
    ldd #"PA
    std word_page
    ldb #'G
    stb word_page+2

% process input spec
    ldx parmsx pointer to it
    lda #1 modeeread
    os? isopen
    bcs error_exit problem opening!
    stx ipath new input path

% get title (if)
    leay title,u
    ldb #40 max size to get

```

```

make.title
    lda ,x+ copy till cr
    cmpa #sd
    beq rw_file ready to print
    sta ,y+ else put in buffer
    decb filled?
    bne make.title

% -----
% Begin read of file
% -----

% enter here start of each page

rw_file
    bsr newpage set new page

% enter here each line of file

rw_line
    leax buffer,u read source line
    ldy #128
    lda ipath
    os? isrdln
    bcs error_exit
    lda #stdout write to std output path
    os? iswrln
    bcs error_exit
    lda lines update line on page
    inca
    cmpa #page.length time new page?
    beq rw_file yes
    sta lines no
    bra rw_line

% exit program with error
error_exit
    cmpb #eof unless end of file
    beq quit ok exit
    lda #stderr say error
    os? fsperr

% exit the program

quit
    clrb set no error
    os? fsexit

% -----
% Subroutine section
% -----

% go top new page, updating:
% 1- date, time
% 2- page #

newpage
    clr lines reset loc on page
    leay date,u get new date
    bsr setdate
    leax number,u point where pa goes
    ldd pageno
    addd #1
    std pageno
    bsr b2d
    ldb #4 max '0' to kill
    bsr kill0s make leading 0's=spaces
    ldd 2,x keep lower-3 digits
    std page & put where belong
    ldb 4,x
    stb page+2
    leax header,u print whole shebang
    bra string remote RTS

% external calls:

ext b2d,setdate,string,kill0s

end start

```

Subroutine: Convert binary number to asc

```
% convert binary # to decimal asc:
% enter: X points to 5-byte array,
% not initialized
% D= value to convert
```

```
% alters: D
```

```
nam b2d
```

```
% first entry: 30Mar84
% updates: none
```

```
% variable storage:
% none
```

```
% tables:
```

```
power fdb 10000,1000,100,10
```

```
% begin code:
```

```
b2d:
    pshs x,y preserve regs
    clr ,s & make local storage
    pshs d save value passed to convert
```

```
% init the ascii array
    ldd #00
    std ,x
    std 2,x
    stb 4,x
```

```
% main loop: get appropriate power
% of 10, see if remaining value of
% passed amount can have this power
% subtracted; loop.
```

```
factor
    leay power,pcr
    ldb 2,s current offset
    aslb
    leay b,y
    ldd ,s check against value left
```

```
% do the (value - 10^4-(2,s))
```

```
subsome
    cmpd ,y can we?
    bcs next.digit no
    subd ,y yes
    inc ,x
    bra subsome
```

```
next.digit
    std ,s save updated #
    inc 2,s update index
    leax 1,x & pointer in ascii array
    lda 2,s done?
    cmpa #4
    bne factor no
    orb ,x should have <=9 left
    stb ,x
    leas 3,s waste local
    puls x,y,pc
```

```
end b2d:
```

Subroutine: Get and Convert Date/Time

```
% get system date
% enter: Y pointing to 17-byte array.
% alters: d,x,y
```

```
nam setdate
```

```
% entry: 30Mar84
```

```
% storage:
number comn 5 for numeric conversions
```

```
% equates:
ftime equ $15
```

```
setdate:
    leas -6,s make space
    tfr s,x time packet
    os? ftime & get it
```

```
% prepare:
    ldb #6 iterations
    pshs b,y count+dest
    tfr x,y source pointer
```

```
% main copy loop
set3no
    leax number,u
    ldb ,y+ source byte
    clra
    pshs y
    ber b2d convert to dec
    puls y
    ldd number+3
    ldx 1,s
    std ,x
    leax 3,x
    stx 1,s
    dec ,s done?
    bne set3no no
```

```
    leas 9,s fix stack
    rts
```

```
end setdate:
```

Subroutine: Print String at X-reg

```
% print string of characters to
% standard output.
```

```
% enter X pointing to string; at ,x is
% length of string
```

```
nam string
```

```
% entry: 30Mar84
```

```
% variables:
% none
```

```
% declares:
iswrit equ $8a
```

```
string:
    ldb ,x+ get count
    clra make 16-bit
    tfr d,y
    inca stdout
    os? iswrit
    rts
```

```
end string:
```

Subroutine: Eliminate Leading 0's

```
% delete leading zeros from an
% ascii decimal string
```

```
% enter: max to do in B
% X points at string
```

```
nam kill0s
```

```

$ entry: 30Mar84

$ variables:
$ none

kill0s:
  pshs b save max
  clrb preset count

killchk
  lda b,x see if '0
  cmpa #'0
  bne killx done if not
  lda #s20 reset to space
  sta b,x
  incb see if maxed
  cmpb ,s
  bne killchk

killx
  puls b,pc fix stack, exit

end kill0s:

OS9 Disk Name Change Utility
$ this program allows for changing
$ the name of an OS9 disk.
$ Calling convention:
$   OS9:name <drive spec> <new name> <RETURN>

$   drive spec is valid RBF drive (/d0, etc)
$   name is 1-32 characters

$ separate spec from name with space or comma
$ New name begins 1st non blank, ends & RETURN

nam name

ttl OS9 Disk-name chang#r
ifpl
type OS9 Disk-name-change Utility
nif

$ first entry: 6April84

$ variable storage

orgv 0

drive.spec var 16 for copy of drv parm
nameptr var 2 ptr to name parameter
opath var 1 path# of drive
buffer var 256 info sector buffer

$ equates:

offset set s1f offset in info sec->name
stderr set 2

$ OS9 function calls:

fsexit equ #06
fsperr equ #0f
isopen equ #84
isseek equ #88
iread equ #89
iswrit equ #8a
iswrln equ #8c

$ strings:

badspec
fcc 'Improper specification',#d

start
leay drive.spec,u cop drv parm->local

drv.chars
lda ,x+ source from parms
cmpa #0d invalid?
beq invalid
cmpa #', separator?

```

```

beq getname yes
cmpa #s20 same
beq getname
sta ,y+ put to dest
bra drv.chars

getname
lda ,x bypass spaces
cmpa #s20
bne gotname
leax 1,x
bra getname

gotname
stx nameptr
ldd #'@<8 for whole drive
std ,y +terminator

$ open whole drive
leax drive.spec,u name
lda #3 for update
os9 isopen
bcs error couldn't
sta opath save #

$ read LSN 0

leax buffer,u where goes
ldy #256
os9 isread do it
bcs error couldn't

leax offset,x advance to name
ldy nameptr & new name
ldb #32 max size of name

movename
lda ,y+
cmpa #s20 invalid?
bcs namend yes, end
sta ,x+
decb filled?
bne movename

namend
cmpb #32 any read?
beq invalid no, bad
lda ,x get last char,
ora #s80 set MS bit
sta ,x

$ seek back to start
ldx #0
pshs u
lda opath
tfr x,u
os9 isseek
puls u

ldy #256 & write back
leax buffer,u
os9 iswrit
bcs error

quit
clrb no error
os9 fsexit

$ bad call on command

invalid
leax badspec,pcr say error
lda #stderr
ldy #100
os9 iswrln
bra quit

$ some disk error encountered

error
lda #stderr
os9 fsperr say error
bra quit

end start

```

RoundOff Errors In The CoCo

Peter A. Stark
Star-Kits Software Systems Corp.
P. O. Box 209
Mt. Kisco, NY 10549

While preparing the second edition of my book on numerical methods, I recently ran across a problem with the numerical accuracy of the Color Computer (or, more specifically, its Microsoft Basic) which may be of interest to other serious users of this computer for scientific or engineering applications. In particular, Color Computer Basic handles numbers in a way which results in some errors which are not normally encountered on other machines.

The very first Program in my book is a very simple demonstration of how a computer generates errors even in trivial problems. It simply adds 0.0001 to itself 10,000 times, and should therefore give an answer of exactly 1. The program is essentially this:

```
1 REM PROGRAM 1
10 S = 0
20 FOR I = 1 TO 10000
30 S = S + .0001
40 NEXT I
50 PRINT S
```

When running this program on an actual computer, we get three kinds of answers:

1. Many computers (including some large mainframes I have tried) give an answer less than 1.
2. Some computers give an exact answer of 1. This generally occurs in those systems which take extra pains to round numbers to their nearest binary equivalent. Computers which use BCD arithmetic (such as 6800 systems using Ultravix Basics) also give an exact answer of 1, as do most programmable calculators.
3. The Color Computer, unlike every other computer I have ever tried, gives an answer of more than 1. In fact, its answer is 1.00000019.

When you consider the reason why errors of this type occur, the fact that the Color Computer gives an answer that is too large is most unusual. If anything, one would expect an answer that is too small, not too large!

Errors of this type are caused by the fact that most decimal fractions (such as .1, .2, .01, and so on) cannot be exactly expressed in the binary number system. For example, even the simple decimal number 0.1 becomes a never-ending binary number

.00011001100110011001100.....

with pairs of ones and zeroes repeating forever. Since a computer cannot possibly hold that infinitely long number, the right end of such a binary fraction is cut off. This creates an error - the number stored is actually smaller than it should be.

That is exactly the point of my sample program. The binary equivalent of the decimal number 0.0001 (in most computers) is actually a bit on the small side, and so adding ten thousand of these will give a sum that is also too small.

Many computers attempt to minimize this problem in a fairly straightforward way. Instead of just cutting off the right end of a binary number when it doesn't fit (this is called 'truncating') and leaving the left end as is, these systems 'round off' the remaining number to the nearest available binary number. (This can be demonstrated with decimal numbers. For example, truncating the decimal number 5.18 to just two digits would give an answer of 5.1, whereas rounding it to the nearest digit would give an answer of 5.2, which is closer to the actual value of 5.18.)

Color Computer Basic also rounds, but it rounds too far! In the above decimal example, this is as if 5.18 were rounded up to 5.3 instead of to 5.2! This leads to a definite error.

In order to see what happens, run the following program. Its purpose is to display exactly how Color Computer Basic stores 0.0001 in binary:

```
1 REM PROGRAM 2
10 LET S=.0001
20 P=VARPTR(S)
30 FOR I = P TO P+4
40 PRINT HEX$(PEEK(I)); " ";
50 NEXT I
```

In this program, we set S equal to 0.0001. In line 20, P is the address in memory where S is stored. The FOR-NEXT loop then prints the five bytes which hold S, in hexadecimal.

Running the program, we get the result

73 51 B7 17 5A

Let us look at this to see what it means. (I assume that by now you are familiar with the relation between binary numbers and hexadecimal numbers.)

Color Computer Basic uses five bytes to store numbers. Of these, the first byte stores the 'characteristic', and the

remaining four store the 'mantissa'. (To allow for fractional numbers, Basic does not store numbers as pure integers, as you might do in assembly language.)

The mantissa is a binary fraction, which is always equal to or greater than a decimal 0.5, but less than 1. (The only exception is that the number zero is stored with a mantissa of 0.) That is, the mantissa is always 'normalized' (except for zero). This means that the first bit is always a 1. Knowing that the first bit of all mantissas is always a one, Color Computer Basic saves a bit of space by omitting this 1 bit and only putting in the remaining bits after that. Instead, the first bit location (the leftmost bit of the second byte of the number) is used to hold the sign of the number.

Hence the bytes 51 B7 17 5A of our 0.0001 give the mantissa as follows:

```
01010001 10110111 00010111 01011010
"....../ "....../ "....../ "....../
: 51      B7      17      5A
:
```

----Sign bit: 0 means plus, 1 would mean minus.

Since the first bit of the mantissa, a 1, is omitted, the actual mantissa is

```
.11010001 10110111 00010111 01011010
```

The characteristic is a power of 2 which should be used to multiply the mantissa to get the correct answer. To allow for positive and negative powers, the characteristic is offset by hexadecimal 80, so that a byte of 80 actually means a characteristic of 0. Thus 81 would be a power of 1, while 7f (one less than 80) is a power of -1.

The largest possible characteristic is FF which (after we subtract out the 80 offset) becomes 7F in hexadecimal, or 127 in decimal. Hence the largest number which can be expressed is approximately

127
2³⁸
which is approximately 1.7 x 10¹¹.

In our example, the leading byte is a 73, which is 13 less than a hexadecimal 80. Hence the power is a -13, and so the above mantissa should be multiplied by

-13
2⁻¹³

In essence, this adds thirteen zeroes in front of the mantissa, with the result that the binary number which represents the decimal number 0.0001 in the Color Computer is a binary

```
.00000000000000 11010001 10110111 00010111 01011010
```

But the correct binary number should actually be

```
.00000000000000 11010001 10110111 00010111 01011000 111....
"....../
32 significant digits
```

We note that 0.0001 in binary is an infinitely long number, with the remaining portion (beginning with the 111....) cut off because the computer can only store the first 32 significant digits of the mantissa.

Now comes the important question: when a binary number is too long to fit into a 32-bit mantissa, should the end of it simply be cut off (truncated)? No, as we have already seen, it should be rounded to the nearest possible binary number.

(Another decimal example may be useful at this point. Suppose we have a decimal number like 3.14159 and want to express it in just five digits. It is obviously better to express it as 3.1416 than 3.1415, since 3.14159 is almost 3.1416 and that makes a smaller error. The rule in decimal is that if the first digit cut off is 4 or less, just truncate, if it is 5 or more, then round by adding a 1 to the last digit being retained.)

In binary numbers, if the first bit of the portion cut off is a 0, then we simply truncate. But if it is a 1, then we add 1 to the next digit to round up. Hence when we shorten the correct answer (A below) to 32 significant digits and round it (because the 111.... portion we cut off begins with a 1), we should get B below:

```
A: .00000000000000 11010001 10110111 00010111 01011000 111..
B: .00000000000000 11010001 10110111 00010111 01011001
"....../
rounded up from a 0 to a 1 -----
truncate -----
```

Hence B above is the best approximation to the decimal 0.0001 that we can express in 32 bits. But now let us compare this with the number as it actually exists in the Color Computer (labelled C below):

```
B: .00000000000000 11010001 10110111 00010111 01011001
C: .00000000000000 11010001 10110111 00010111 01011010
```

NOTE WELL: The correct (rounded) binary number ends with 001, while the Color Computer's version ends with 010. The Color Computer has obviously rounded up by an extra one in the 45th place; hence its 0.0001 is about

too big! This is a mistake, and thus adding this number to itself 10,000 times is guaranteed to result in a sum which is too large. (Actually, even if we use a POKE to fix the number, the answer still comes out wrong because the same rounding process is used after each addition, and so the error is still wide.)

The problem is apparently an error in Basic's rounding routine. We don't even need a very fancy program to see this error - just try something like PRINT 4.102, and you will see that the computer prints the answer as 4.1020000 instead. The problem is again due to a rounding error.

Basic seems to use the same rounding subroutine not just after a conversion from decimal to binary, but also after arithmetic operations. If you want to experiment some more to see how Color Computer Basic makes such errors, here are a few simple Basic lines to try:

```
PRINT 0.0001*10000 - I should give you an answer of 0.
Instead, you will get 4.65661287E-10, which just happens to be equal to
```

-32

This indicates that the binary answer, instead of being just 1.0000..., is in fact 1.00000000000000000000000000000001, which is an error in the 32nd bit.

Try PRINT 4.102/1 and you get the right answer, but try PRINT 4.102/1*1 and you get an error.

Another way of demonstrating the same effect is to substitute the line
10 LET S = 0.0001 * 10000
in program 2 above. It will then output the result

```
01 00 00 00 01
```

which also indicates that the rightmost bit of the 32-bit mantissa is a 1.

Small errors in the handling of fractional numbers are unavoidable and normal in binary computers; they are an inherent part of the way in which binary computers handle so-called 'floating-point' numbers. Experienced programmers anticipate these and work around them. But the rounding routine in Color Computer Basic is downright wrong - and potentially dangerous because it introduces errors of a type not normally present, and hence not anticipated by even experienced programmers.

1112 College Ave
Regina, Saskatchewan, Canada
S4P 1A8
February 13, 1985
Don Williams

As a subscriber to 68 Micro Journal since 1978, I have noticed subtle changes in the journal and the rise and fall of Color Micro Journal. These changes must come about due to advancing technology and the desire of the readers to keep abreast of new developments.

I read everything in 68 Micro Journal. My direct needs are support for the Flex system on a 6809 S550 system using XBASIC, TSC Assembler, TSC Editor, TSC Sort Merge and TSC Text Processor. The educational needs are for OS-9 on the CoCo. Stylograph and Dynacalc run on the CoCo. I have accounting packages to run on both machines. I suspect that I am typical of many readers and cannot use much of the information printed. However by reading through, a little bit rubs off and sinks in to my thought processes. I would like your readers to understand that your bit bucket prints what you receive. Many of us have solved what to us was a tricky problem in software or hardware. The bit bucket can be considered as our user group newsletter and it is up to us to contribute.

It is over 2 years since you printed anything from me so here is my contribution. The TSC assembler furnished with my Gimix Flex printed a string of zeros following the date. Examination of the object code revealed that the assembler was looking for the 58167 hardware clock on the Gimix CPU. My CPU was SWTPC and my hardware clock was a JP7 using a 5832. A disk utility that printed the time on the terminal was modified to write the time into Flex scratch and the assembler was patched to look at the Flex scratch rather than the 58167. If the clock was read for each header there would be a different time on each page. This method requires that the clock must be polled before the assembler is used. A key factor was that the 5832 software presents the time digits in reverse order to the 58167 registers.

The software for the JP7 has been printed in 68 Micro before. Anyone desiring help on this should send a formatted 5" disk 35 or 40 trk SS or DS along with \$2.00 for Canadian postage. Please include some public domain software on the disk. Also a text file describing your interests and your system.

Weldy Moffatt

Damon Hill
3261 Circle Oak Dr. NW
Atlanta, GA 30339

Dear Sirs:

I was enjoying your January 1985 issue of Color Micro Journal, especially Carl Mann's column on page 26 because I use Telewriter so much. Unfortunately, a good portion of that page and page 7 were obscured by a bad printing job and I'd like to at least get a copy of those pages.

As Carl Mann probably knows, it's also possible to use Telewriter's loader program to POKE in 6ms stepping and transferring drives 2 and 3 to the back side of double-sided drives. Until I installed ADOS, that was my standard startup procedure, which gave me a welcome increase in disk storage and a convenient means of backing up text files. The exact POKEs depend on the version of Disk Basic one is using.

I'm very sorry to see that you've discontinued publication of CMJ. I looked forward to reading Mann's column and the Basic09 column. While 68MJ is of at least some interest to me, I may not renew my subscription if you completely discontinue coverage of the Color Computer.

Like a lot of owners, I can't really afford to buy a Gimix or one of the other big SS-50 systems, the CoCo was and is the best way for those with a limited budget to get into computing cheaply and effectively. If we can't get good coverage on hardware and software, it's going to be difficult to make the transition from games to programming and serious applications. RAINBOW and UNDERCOLOR provide some of that coverage, but more is needed.

Some day I hope to get a larger and more capable computer, probably a single-board design and hopefully 68000-based with multi-megabyte memory capability, hard disk and greatly improved graphics, the latter being one of my interests. The Macintosh is NOT what I want in a computer, however. Too expensive, too limited in design and expansion capability and generally just too "Apple".

Your coverage of single-board computers in recent issues of 68MJ interests me, and I'd like to get the related back issues prior to the March '85 issue if you'll let me know how much they are.

Damon Hill

20426 Lichfield
Detroit, MI 48221

Don Williams, Publisher
'68 MICRO JOURNAL
Hixson, TN 37343

Dear Don,

I've been a long-time reader (since your first issue) and sometime contributor to MICRO JOURNAL, and I enjoy (along with about everything else I find there) your editorials. You sound like a dyed-in-the-wool Motorola fan, probably like a lot of your readers. I found your ravings in the November issue particularly inspiring. I always knew I picked the right machine back in '77 (SWTPC 6800), but it's nice to hear someone else holler about it.

I had an inspirational (?) experience recently that I'd like to share. It'll probably have the same effect on you that it did on me.

I work as a technician for a large telecommunications company that was once part of a lot bigger company, if you know what I mean. They recently sent me to a two-day school to learn about a piece of ancillary equipment that we will soon be responsible for. It's made by a company called Timeplex, and it's called a Statistical Time Division Multiplexer. Its basic function is to take up to 24 channels of serial (RS-232C) data, and multiplex them onto two 2-wire data links. It comes in a box that looks something like a

home computer. Inside, it has a motherboard and bus architecture, with seven slots for plug-in 'modules'. One of the modules is called the control module, and it controls up to six other modules, each of which carries four channels of data. The control module has a four-digit seven-segment LED display, four thumbwheel switches (with hex digits) and two switches for programming. From this 'front panel', you can program any parameter of any channel-baud rates from 50 to 19,200, number of stop bits, parity, synchronous/asynchronous operation, number of data bits, etc. It's programmable to accept ASCII, EBCDIC, or Baudot code. You can also run diagnostics, which report the suspected trouble TO THE COMPONENT LEVEL. I mean, it will tell you which IC to change! Of course, on power-up, the main module automatically diagnoses its half-dozen major subsystems and reports the results on the LED display. The parameters are stored in a battery-backed-up RAM that will maintain its data for 90 days, so you don't have to re-program everything if you power the whole system down. There's also an optional 'supervisory port' on the control module which allows you to plug in a CRT, keyboard and printer. With the keyboard, you can use an extensive built-in 'command language', which allows you to do everything the thumbwheel-switches allow and much more. Such as obtaining histogram-type information about traffic, errors, etc. on any channel or either data link. Any channel of data can be explicitly routed through either of the data links, or all may be routed through one link. It also has an optional feature called 'traffic balancing' where it will switch channels between the two links to balance the traffic. If these multiplexers are used at both nodes (both the transmitting and the receiving end, usually miles apart) the built-in diagnostic facilities allow you to 'loop' the circuit both locally AND at the far end. This is used to eliminate certain stages of equipment in searching for troubles.

The instructor for the class (who works for the same company I do, not the vendor) was ecstatic about this little box. He rightly attributed its tremendous power and versatility to its microprocessor-based hardware.

The hardware? The control module has about 16K of 2716 EPROM and 16K of 4116-type dynamic RAM. Each channel module has 8K of onboard RAM for buffering. Which channels use how much of the buffer under what circumstances is also programmable (from the main module, of course!)

Oh yes, the control module and each channel module has its own microprocessor. A 68000 you say? An 8086? A Z-80? No. Each board has one 11' 01' 2-MHz 6800 (a 68800). Not a 6809. A 6800. It's also chock full of 6821 PIAs and uses 6854 Data Link Controllers, too.

Strangely, the documentation is also a technician/user's dream. Is this something that automatically happens to products built around Motorola microprocessors? The manual was written as if they were damned proud of this thing and they want to make sure you know all about how to use it and program it, so you'll see how fantastic it is too!

Yours,



Keith Alexander

Letter from:
Technical Systems Consultants, Inc.

February 28, 1985

Dear Don,

I have noticed that "68 Micro" has been starting to cover 68000 related topics so I thought I would pass along some information your readers might find interesting. As you recall, Technical

Systems Consultants designed UniFLEX(tm) for the 68000 but the original implementation was on the 6809 since the 68000 was not yet available. We started the 68000 version in August 1981 and had it fully operational in July of 1982. Since that date, we have continually improved the system and have developed a full line of support software. When the 68010 became available, we developed a demand page, virtual memory version of UniFLEX. This version has been fully operational since June of 1984. I would like to point out that the primary difference between the 68000 and 68010 is that the 68010 is designed to support a virtual memory environment which the 68000 can't. UniFLEX is currently the only available operating system available from a software house which runs in a demand page, virtual memory mode on the 68010 and 68020. (There is one other available, Berkley 4.2 Unix(tm), available "as is" from the university.)

The speed and efficiency of UniFLEX running on a 68010 is absolutely incredible, but the real winning combination is UniFLEX and the 68020. The performance is awesome! We currently have UniFLEX running on several different 68020 systems in house and are in the process of rewriting every module of the kernel to take advantage of the extended instruction set and addressing modes of the 68020. Very shortly I will be sending you some kernel timings comparing the 68000, 68010, and 68020 versions of UniFLEX so you can see for yourself.

The kernel size for the virtual memory version of the system is quite compact considering its complexity. It resides in approximately 36K of memory. We have done scaled down versions for process control which will run in as little as 8K of ROM. A lot of people don't realize the versatility of UniFLEX but we have done many dedicated, scaled down versions of the system. As an example, it is the controlling software in the SWTPC PPI board and also in the GIMIX IOP. Both of these are small ROM based versions of UniFLEX!

We have added a large number of features to 680X0 UniFLEX including new system calls, and an expanded standard set of utilities. The list of added features is too long to describe in detail but a close examination of the enclosed UniFLEX manual will allow you to see for yourself. Besides the standard utilities, we have a full line of support software including Extended BASIC, BASIC Precompiler, COBOL, FORTRAN 77, C, Relocating Assembler, Linking Loader, and Sort/Merge. What's most important, these products have been running on the 68000 for quite some time (some as long as 3 years) and are available now. In the works are the Enhanced Printer Spooler (probably available by the time this is read) and an ANSI Standard Pascal Compiler.

In addition to our product line, there has been considerable support from outside vendors in the way of major product development for UniFLEX. To name a few, Franz Inc. has ported their Franz Lisp (a

full implementation of Common Lisp for use in Artificial Intelligence), Logicware has ported Prolog (again for the AI community), and Network Research Corp. is in the process of porting their extremely powerful networking package. Smalltalk 80 is available under UniFLEX on the Tektronix 4404. In development (by a company we can't disclose at this time) is a DOD verified ADA Compiler which should be available in the near future. As you can see, this list is quite extensive, but it is only a partial list. We have quite a few other development projects going on, both in and out of house, which aren't quite ready to be revealed to the public. We will keep you informed.

One product of particular interest is our C compiler. Developed in-house, specifically for the 68000, it is an extremely efficient compiler. We actually have two C compilers, one for the 68010 (and 68000), and one for the 68020. The 68020 compiler takes full advantage of the additional addressing modes as well as the added instructions (i.e. for bit fields, etc.). Both compilers are full Unix System V compatible. The System V compiler has several extensions over the "Kernighan & Ritchie" definition, including enumerations, structure assignments, passing structures to functions, returning structures from functions, void types, and fully qualified structure and union members. The C Library, also System V compatible, is very extensive. I have also enclosed the C compiler manual (as large as the entire UniFLEX manual) so you may see for yourself the completeness of this product.

If you would like to try out 680X0 UniFLEX, there are currently several systems available (with many more over the next several months). We have a wide variety of systems available for various VME based systems, including Motorola, Signetics, and Force card based systems, as well as a version for the Motorola VME-10 system. There are versions also available for Tektronix, Pixel, and Momentum machines. The Tektronix 4404 is a good example of how UniFLEX has been expanded to support bit-mapped graphics, a mouse, and other state of the art hardware. Tektronix is selling the 4404 as a personal AI work station, but since the system comes with our C compiler and full utility set, it is a great development system. GIMIX has announced their 68000 system which will of course run UniFLEX. We are currently finishing four ports to a variety of hardware which we will be announcing to the world shortly.

Several of our support products are also available to run on a variety of Unix or Xenix(tm) based systems and more are becoming available almost weekly. One example is our Extended BASIC and Sort/Merge for the Tandy 6000 (and Model 16). Since our BASIC is written in assembler language, it runs 5 to 7 times faster than the BASIC available from Tandy. If you have a need for any of our products on a 68000 based Unix system, give us a call. We may have it or be working on it.

Hopefully this letter has been informative to everyone. It should bring you up to date as far as 680X0 matters are concerned. One last note. We are still developing new products for the 6809. We have two new products we are quite excited about and they are almost ready to release. You should be hearing from us very shortly about these.

Sincerely,

Dave Shirk

UniFLEX registered in U.S. Patent and Trademark Office.
Unix is a trademark of AT&T Bell Laboratories.
Xenix is a trademark of Microsoft.

VIRTUAL MEMORY OPERATING SYSTEM

TAKES FULL ADVANTAGE OF THE MOTOROLA 68010 and 68020

Chapel Hill -- Technical Systems Consultants, Inc., has released a virtual memory version of its UniFLEX® Operating System which is specifically optimized for the architectures of the Motorola 68010 and 68020 microprocessors. The operating system presently runs on the Motorola MVME120/050, the Motorola VME10, and FORCE Computers' CPU-3 system. Versions for other manufacturer's VME hardware are planned.

This release is the culmination of years of research and development into software designed exclusively for the Motorola 68xx and 68xxx family of microprocessors. Although the UniFLEX system offers most of the features found in UNIX systems, its performance is dramatically better due to an assembly language implementation and specific optimization for the Motorola 68xx family.

The UniFLEX system is the only currently available operating environment specifically designed for the 68xx which may be adapted to systems ranging from ROM based applications, to single-user, multi-tasking workstations, to full multi-user, multi-tasking, time-sharing systems. A typical kernel, without any device drivers will reside in approximately 26K of memory.

Technical Systems Consultants, Inc. was formed in 1976 and specializes in operating systems, languages, and utilities for the Motorola 68xx and 68xxx family of microprocessors. The multi-user, multi-tasking, UniFLEX Operating System has been running on 6809 based hardware since September, 1980, and on 68000 based hardware since October, 1982. Further information may be obtained from Technical Systems Consultants, Inc., 111 Providence Road, Chapel Hill, NC 27514 919-493-1451 or TVA 510-920-0540 TSC CPED.

*UNIX is a trademark of AT&T/Bell Laboratories.

*Ethernet is a trademark of Xerox Corporation

*UniFLEX Registered in the U.S. Patent and Trademark Office

MICRO INTERNATIONAL

Industrial Microcomputer Systems

BOX 47 • EAST FAIRFIELD, VERMONT 05448 • 802-827-3827

-For Immediate Release-

Micro International announces the availability of a new mathematical library which is compatible with the following C compilers LATTICE, MICROSOFT, CB6, DESMET, INTEL, and McCOSH/MICROWARE compilers under PC DOS, MS DOS, FLEX and OS-9. The source code is included for CORDIC and POLYNOMIAL techniques. Price of this package is \$100 and the contact person is KARL R. ZURAV, MICRO INTERNATIONAL, BOX 47, EAST FAIRFIELD, VERMONT 05448 802-827-3827.

Product Description Sheet Attached.

MICRO INTERNATIONAL

PC-8088/8086 SYSTEM
BOX 67, EAST HAVEN, CT 06424-0067

MICRO INTERNATIONAL C-LANGUAGE MATHEMATICS LIBRARY

The MICRO INTERNATIONAL C-MATHEMATICS LIBRARY consists of:

1. An include file, MATH.H, which defines a number of useful mathematical constants and defines the functions in the mathematical library as returning floats.
 2. A library of mathematical functions, LIBMATHC.R, generated using CORDIC techniques.
 3. A library of mathematical functions, LIBMATHC.P, generated using polynomial approximations.
- A listing of MATH.H and the contents of the libraries, LIBMATHC, and LIBMATHP are included in the appendix.

The file TEST.C is a simple test program which demonstrates the use and accuracy of the functions. Two sample outputs are provided:

TESTC.ANS generated using the TEST.C program and the CORDIC library LIBMATHC
TESTP.ANS generated using the TEST.C program and the LIBMATHP library.

In general, the polynomial approximations have an accuracy of 6-7 digits. The CORDIC approximations, which have a speed advantage over the polynomial approximations, show an accuracy of approximately 5-6 digits. As usual, the trigonometric functions accept angular arguments expressed in radians.

The source files for both the CORDIC and polynomial libraries are included on the disc, along with several assembly language support routines.

The C-Language Mathematics Library is available under the following operating systems: PC DOS, MS DOS, FLEX and OS-9 for use with the LITTEC, MICROSOFT, C86, DESMET, INTROL and MCCOSH/MICROWARE compilers.

Dear Don,

Readers of '68' MJ who still enjoy wielding a soldering iron and would like to add graphics to their S-50 bus '09 system, might be interested in a design of mine which I am willing to make available to fellow enthusiasts. Briefly, the board is a simple bit-mapped, single-plane (B&W) video graphics generator with 16K of screen RAM (2 x 6264) giving 512 x 256 pixels. It is based on a 6845. Screen refresh is "transparent"; i.e. the CPU is not retarded and there is no disturbance to the display during CPU access of VRAM. Extended addressing is supported.

Actually, the main reason for developing the board was to improve my "C" programming ability. Apart from a small "core" of low-level drivers in machine code, the graphics support routines (symbol and line drawing, etc) are written in "C".

I would be pleased to receive enquiries. If there is sufficient interest shown, I could be persuaded to get a quantity of boards fabricated for distribution at minimal cost. (Enquiries from potential US distributors also welcome.)

Yours sincerely,

Michael J. Bauer

Michael J. Bauer
P.O. Box 221
Ivanhoe
Victoria, 3079
AUSTRALIA

Dear Don,

Could you please send me information about the Windrush Screeneditor III that S.E. Media sells.

Thanks very much. I am new to the 68XX fraternity, having picked up my 6800 (40K), MF-68, and CT-82 at a local garage sale for \$30.00!!! It took a while to locate your Journal - it has been a lot of help thus far, even though most of the articles deal with the 6809.

Cordially,

Craig E. Henrikson

Craig E. Henrikson, Ph.D.
37 Herrick Avenue
Delmar, NY 12054

1810 N.E. Fremont
Portland, OR 97212

Dear Sirs;

Concerning E. M. Pass's enhancement of SWTPC's MIRROR.CMD, you left out a 'b's xloop' before 'restor ldb trgrdv' and I think the four lines before 'chks16 lda maxsec' go before 'chks12 tst single'. You are saving the Target SIR and replacing it after the mirroring is complete.

Also I modified TSC's PR.CMD to handle superscripts and subscripts and would be happy to share same if it's of interest to your readers.

Also, why is it that your subscription rate less as to go from 1 year to 2 years than from 2 years to 3?

Yours truly,

Gary Lemoine

Gary Lemoine

4522 N. 22 Street
Milwaukee, WI. 53209

68 Micro Journal
5900 Cassandra Smith
P.O. Box 849
Hixson, TN 37343

Help:

I have a problem which I hope some of my fellow 68 Micro readers may be able to assist me with. My Gimix system is great, works well and is very useful. But, using the Gimix video board and drivers for terminal emulation leaves a lot to be desired. Using a PIA for keyboard input and the primitive "terminal-like" functions is very limiting.

Surely others out there have felt this problem, and perhaps solved it by rewriting the software. I would sure like to hear from anyone willing to share their experience with me.

thank you for your time.

Sincerely,

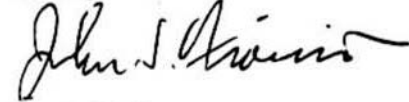
Jim Bertrand

Jim Bertrand
WA9CXG

Gentlemen:

Has anyone interfaced the following printers with the SWTPC 6800 CPU with a serial port, 1st choice, Paralled port 2nd choice. Printers are: Radio Shack, Gorilla/Banana, Star Gemini-10X, Okidata, Epson. Would appreciate any help. Thanks.

Sincerely,



John J. Florino
518 - 85th Street
Brooklyn, N.Y. 11209

WESTCHESTER Applied Business Systems
2 Pea Pond Lane, Briarcliff Manor, N.Y. 10510

February 21, 1985

TO: Bob May
RE: SWTPC P.CMD

Dear Bob,

When we first purchased our SWTPC computer back in '79, we never could get the printer driver (P.CMD) to work with our Centronics 737 printer. Basically, the computer was not acknowledging the BUSY lead, and would hang up after one character was sent. As a result, we wrote a driver with timing loops for characters and carriage return, the latter based upon number of characters sent. We've been using this driver for six years, which is somewhat slow, but adequate.

Recent mechanical difficulty with the 737 carriage return, however, has caused overwrites and lost data in this unacknowledged method, and prompted us to invest in an Epson FX-80. (Thanks for the recommendation and prompt delivery!) Thus, we reinvestigated the P.CMD issue - This time a little older and wiser!

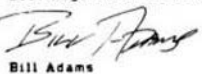
As it turns out, we missed a subtle point: The SWTPC documentation for the MP-LA board shows a BUSY lead indication with a little "bar" over the word "BUSY". This of course means "Inverted BUSY" or "NOT BUSY". Apparently the P.CMD illustrated was written before parallel printer standards were adopted, since most printers use a non-inverted BUSY.

[Further study of the documentation revealed some other interesting points such as the fact that the computer sees the ready indication as a result of a TRANSITION of the busy lead - Something we, in '79 did not understand from the documentation at the time.]

With minor modifications to the bit patterns stored into the control register (inverted 2-bit) we were delighted to see the Epson come to life and literally pour out information in perfect form. Further, the same driver worked perfectly with 737 - with about a 20% increase in speed despite occasional slow head returns!

A listing is enclosed - some of the original labels are renamed due to many coding iterations, but it is essentially the same as the SWTPC driver. The major changes are circled. You may wish to drop this in the "Bit Bucket" section for those who have had similar problems.

Best Regards to All at CPI,


Bill Adams

P.S. This was printed with the revised driver on the 737 - The listing was done on the Epson using the same. The Epson is about three times faster, enhancing our programming effort on XDMS-IV (I had to get it in!

UTILITY #1120 P.OUT - 02-21-85 PAGE 1

```
*** P.CMD - PARALLEL PRINTER DRIVER
* 01/21/85 by Bill Adams
* For configurations using:
* CA1 = BUSY lead (Non-Inverted)
* CA2 = STROBE (Inverted)

C300 0000 0000 0000 0000 0000 0000 0000
C301 0000 0000 0000 0000 0000 0000 0000
C302 16 000F INIT LBR4 INIT1 -ENTRY VECTORS
C303 16 001F CLOS LBR4 CLOS1
C304 16 001F PUTC LBR4 PUTC1
C305 16 0037 PCW9 LBR4 PCW91

C306 E01C PORT FDB E01C -Default Port [7]
C310 00 SIDE FCB 0
C311 00 RSVD FCB 0
C312 00 FLAG FCB 0FF
C313 00 CNTR FCB 0

*** INIT1 - INITIALIZE PIA & PRINTER
INIT1 PWS 1A
C314 AE 0C FS LDA LBR4 PCW9 -INIT PIA
C315 00 30 LDA LBR4 PCW9 -BEL DORS
C316 A7 01 STA LBR4 PCW9 -BEL OUTPUT
C317 00 00 LDA LBR4 PCW9 -BEL DATA REGS
C318 A7 01 STA LBR4 PCW9
C319 00 00 LDA LBR4 PCW9
C320 00 00 STA LBR4 PCW9
C321 00 00 LDA LBR4 PCW9
C322 A7 01 STA LBR4 PCW9
C323 00 00 LDA LBR4 PCW9
C324 00 00 STA LBR4 PCW9
C325 00 00 LDA LBR4 PCW9
C326 00 00 STA LBR4 PCW9
C327 00 00 LDA LBR4 PCW9
C328 00 00 STA LBR4 PCW9
C329 00 00 LDA LBR4 PCW9
C330 00 00 STA LBR4 PCW9
C331 00 00 LDA LBR4 PCW9
C332 00 00 STA LBR4 PCW9
C333 00 00 LDA LBR4 PCW9
C334 00 00 STA LBR4 PCW9
C335 00 00 LDA LBR4 PCW9
C336 00 00 STA LBR4 PCW9
C337 00 00 LDA LBR4 PCW9
C338 00 00 STA LBR4 PCW9
C339 00 00 LDA LBR4 PCW9
C340 00 00 STA LBR4 PCW9
C341 00 00 LDA LBR4 PCW9
C342 00 00 STA LBR4 PCW9
C343 00 00 LDA LBR4 PCW9
C344 00 00 STA LBR4 PCW9
C345 00 00 LDA LBR4 PCW9
C346 00 00 STA LBR4 PCW9
C347 00 00 LDA LBR4 PCW9
C348 00 00 STA LBR4 PCW9
C349 00 00 LDA LBR4 PCW9
C350 00 00 STA LBR4 PCW9
C351 00 00 LDA LBR4 PCW9
C352 00 00 STA LBR4 PCW9
C353 00 00 LDA LBR4 PCW9
C354 00 00 STA LBR4 PCW9
C355 00 00 LDA LBR4 PCW9
C356 00 00 STA LBR4 PCW9
C357 00 00 LDA LBR4 PCW9
C358 00 00 STA LBR4 PCW9
C359 00 00 LDA LBR4 PCW9
C360 00 00 STA LBR4 PCW9
C361 00 00 LDA LBR4 PCW9
C362 00 00 STA LBR4 PCW9
C363 00 00 LDA LBR4 PCW9
C364 00 00 STA LBR4 PCW9
C365 00 00 LDA LBR4 PCW9
C366 00 00 STA LBR4 PCW9
C367 00 00 LDA LBR4 PCW9
C368 00 00 STA LBR4 PCW9
C369 00 00 LDA LBR4 PCW9
C370 00 00 STA LBR4 PCW9
C371 00 00 LDA LBR4 PCW9
C372 00 00 STA LBR4 PCW9
C373 00 00 LDA LBR4 PCW9
C374 00 00 STA LBR4 PCW9
C375 00 00 LDA LBR4 PCW9
C376 00 00 STA LBR4 PCW9
C377 00 00 LDA LBR4 PCW9
C378 00 00 STA LBR4 PCW9
C379 00 00 LDA LBR4 PCW9
C380 00 00 STA LBR4 PCW9
C381 00 00 LDA LBR4 PCW9
C382 00 00 STA LBR4 PCW9
C383 00 00 LDA LBR4 PCW9
C384 00 00 STA LBR4 PCW9
C385 00 00 LDA LBR4 PCW9
C386 00 00 STA LBR4 PCW9
C387 00 00 LDA LBR4 PCW9
C388 00 00 STA LBR4 PCW9
C389 00 00 LDA LBR4 PCW9
C390 00 00 STA LBR4 PCW9
C391 00 00 LDA LBR4 PCW9
C392 00 00 STA LBR4 PCW9
C393 00 00 LDA LBR4 PCW9
C394 00 00 STA LBR4 PCW9
C395 00 00 LDA LBR4 PCW9
C396 00 00 STA LBR4 PCW9
C397 00 00 LDA LBR4 PCW9
C398 00 00 STA LBR4 PCW9
C399 00 00 LDA LBR4 PCW9
C400 00 00 STA LBR4 PCW9
```

```
*** CLOS1 - O/P CR
CLOS1 LDA #00D

*** PUTC1 - PRINT A-REG CHAR
PUTC1 BSR PCHK1
      BPL PUTC1
      PSMB X,B,A -PUT CHAR
      LDX PORT,PCR
      STA 0,X
      CLR FLAG,PCR
      BSR PCHK1
      LDA #034 -STROBE LO
      STA 1,X
      BSR PCHK1 -WAIT
      LDA #03C -STROBE HI
      STA 1,X
      PULB X,B,A,PC

*** PCHK1 - CHECK FLAG/PORT
PCHK1 PSMB X
      TST FLAG,PCR
      BNE PCHK9 -READY
```

UTILITY #1120 P.OUT - 02-21-85 PAGE 2

```
C 4C AE 0C BF LDX PORT,PCR
C34F 00 01 TST 1,X
C351 2A 07 BPL PCHK9 -NOT READY
C353 00 04 TST 0,X
C355 00 05 BSR RTN -BUSY-ACKNLB WAIT
C357 03 0C AR COM FLAG,PCR
C35A 35 90 PCW9 PULB X,PC

C35C 39 RTN RTS
C35D 00 ENDS FCB 0
END
```

0 ERROR(S) DETECTED

SYMBOL TABLE:

CLOS	C305	CLOS1	C327	CNTR	C313	ENDS	C350	FLAG	C312
INIT	C302	INIT1	C314	PCW9	C308	PCHK1	C345	PCW9	C35A
PORT	C30E	PU C	C308	PUTC1	C329	RSVD	C311	RTN	C35C
SIDE	C310	SIDE	C300						

Svein Solstad
Thorstein Stenbaksvei 80
N-3600 Kongsberg
NORWAY

I have had the pleasure to read your magazine for a few months, and I like what I have read. But there are things which I would like to see more about. I am thinking of 68000 based machines which are running some kind of UNIX. There was an article in the January issue which I found interesting, namely a short review of UNIX. If you can print more articles of that kind, I would be very pleased.

I do have such a machine, based on the VME-bus (Europe) with a 68000 CPU and the necessary RAM, timer, serialports, floppy and winchester. It is running one of the UNIX lookalikes, namely IORIS. This operating system is delivered by Whitesmith, and is easily adapted on most 68000 based machines. There is not need for a memory management system as for most UNIX systems.

IORIS is a small system, less than 2 Mbytes, and the need for large disks is not present. Actually, the first time I started the system, I had only 2 floppydrives, 820Kbytes each, and it was to live with. I had to split the system into three different discs, and change between them and my own datediscs when necessary. But of course, the winchester, 5MBbytes, was ordered within an year. But since IORIS is that small, there are some limitations in it, compared with other UNIX systems. But the most useful stuff is there. But the nice things like sed, awk, lex, yacc and make is not there. But if you are used to play with UNIX, you will feel yourself at home.

The system has possibilities for compiling programs written in Pascal and C. The compiler is powerful with possibility to tune it for your specific needs. There is also an assembler as part of the compiler system, and a linker of course. The Pascal is very limited, it lacks all the nice extensions which makes the language useful. That is not the case with the C, which does not lack anything, but differs a little from other C-compilers, but it is absolutely to live with.

One thing which is not supplied with the system, is a screen editor. The standard line editor is there of course, but very soon, you want to have a screen editor to play with. To solve this problem, I got a screen editor from U.K., which was named Thier (I did not stole it!). This is an EMACS-like editor, and very powerful. I have played with a few other screen editors, but none of them can beat this one. You maybe wonder about the strange name, just as I did! It is short form for "This Here Isn't Even Find", and Find is another screen editor. There is only one function this editor lacks! The possibility to define a rectangular part of the screen and copy this to anywhere else in the file or on the screen. The editor has also simple text formatting: left and right adjustment.

There is one problem I do have, there is very few, actually none, software packages, which execute under IORIS, available here in Norway. Whitesmith has sent out a short list of things they claim are available, but I guess there are more out there. I am especially interested in databases, calculation programs and text formatters. If there are anyone out there who knows about sources of such stuff, please tell me. I can't see any reason to make this tools myself, when they do exist somewhere.

Another thing which might interest you, is a short description of the hardware I use. It is a singleboard system, where everything resides on the board, except the power supply, floppy and winchester with the

controller. On board you will find the 68000, 512KByte DRAM, 128KByte EPROM, two serial ports, timer, parallel port and the VME interface. The latter is not used now, but will be when extending the system. size of the board is small, 160mm by 235mm. standard double Euroboard. So together with 3.5inch drives and a switching power supply, it becomes a very compact "UNIX" machine, nonexpensive too (about \$2700 here in Norway, but we have higher prices on hardware than in U.S.).

If there are people out there, who would like to know more about this homebrew project, just send a note to the address below and information will be sent you. The PCB is available together with the documentation and license.

Well, now you know about a system which differs some from what most of you are familiar with.

SSo.

Information address: Byvind Raa
Veungadelsveien 1
N-3800 Kongeberg
Norway

43 COBK ROAD,
BRYLANDS,
CAPB TOWN, 7700
SOUTH AFRICA,

678 FEBRUARY 1985

DEAR DOK,

PLEASE FIND ENCLOSED A RENAMED VERSION OF NICO YSSEL'S LOG UTILITY AS PUBLISHED IN THE NOVEMBER #4 ISSUE OF 68 MICRO JOURNAL. I WOULD ALSO LIKE TO BRING TO YOUR ATTENTION A PROBLEM WITH 86M SLAGHEKK'S DO UTILITY AS PUBLISHED IN THE JANUARY #5 ISSUE. THE UTILITY ASSUMES THAT THE Y INDEX REGISTER IS NOT DISTURBED BY THE UTILITY THAT IS CALLED BY FLEX'S 'DOCKND'. TO CORRECT THIS PROBLEM THE Y INDEX SHOULD BE PRESERVED BEFORE CALLING 'DOCKND' AND RESTORED UPON ITS RETURN. IE.

PSHS Y
JNR DOCKND
PU S Y

RON ANDERSON MENTIONED THAT PETER STARK HAS USED A BYTE TO RECORD THE TIME THAT A FILE WAS WRITTEN TO DISK. IN THE INTERESTS OF STANDARDS WHICH YTS WAS USED? RON ALSO MENTIONED VARIANTS OF THE FLEX FILE SPEC. SYNTAX. HE DID NOT MENTION A FURTHER VARIANT WHICH IS ALSO VALID. AND THAT IS FILENAME.DRIVE NUMBER OR DRIVE NUMBER.FILENAME WHERE THE DEFAULT EXTENSION IS USED. SG.

***LIST LETTER.2 WHERE THE DEFAULT EXTENSION IS .TXT

I WOULD LIKE TO WISH 68 MICRO JOURNAL EVERY SUCCESS FOR THE FUTURE. BYTS LOOKS LIKE A JOURNAL FOR I&X AND ITS CLONES THESE DAYS.

YOURS FAITHFULLY,

John Ritchie
JOHN RITCHIE

- LOG.CMD REVISED
- SEE 68 MICRO JOURNAL NOVEMBER 1984, PAGE 32
- THE PUBLISHED PROGRAM HAS SEVERAL PROBLEMS.
- 1. THE FLEX PROMPT LOCATION IS DIFFERENT FOR THE GENERAL VERSION OF FLEX.
- 2. IF AN ATTEMPT IS MADE TO OPEN A FILE THAT ALREADY EXISTS THE PROGRAM CRASHES.
- 3. MEMEND NOT RESTORED TO ITS FORMER VALUE.
- THE FOLLOWING CHANGES HAVE BEEN MADE.
- 1. THE LOG FILE THAT IS OPENED DEFAULTS TO THE WORK DRIVE.
- 2. THE BLOCK TRANSFER ROUTINE
- 3. THE NEW VECTORS FOR INCH2, OUTCH2, FMSCLS AND MEMEND ARE ONLY INSTALLED IF THE OPENING OF THE LOG FILE IS SUCCESSFUL.
- 4. ONLY THE SAVED VECTORS, THE NEW INCH, OUTCH, FMSCLS ROUTINES AND THE LOG FILE FCB ARE RELOCATED.
- 5. THE ABOVE PROBLEMS FIXED.
- AN UNDOCUMENTED FLEX VARIABLE IS SAVED AND RESTORED. IT IS LABELED RETADR, LOCATED AT 00C43/4.
- WOULD SOMEONE ENLIGHTEN ME AS TO WHAT IT IS ?
- SEE ORIGINAL PROGRAM LISTING FOR COMMENTS.
- THIS CODE WAS DERIVED FROM A DISASSEMBLY.
- EXTERNAL LABEL EQUATES

```
000C FCBEXT EQU 12
000D DEFDRV EQU 0CC0D
002D GETFIL EQU 0CD2D
002B MEMEND EQU 0CC2B
0043 RETADR EQU 0CC43
004E PROMPT EQU 0CC4E
0003 WARMS EQU 0CD03
000C INC 2 EQU 0CD0C
0012 OUTCH2 EQU 0CD12
001E PSTARR EQU 0CD1E
003F PSTERR EQU 0CD3F
0043 FMSCLS EQU 0D403
0046 FMS EQU 0D406
0049 FCBPTR EQU 0D409
```

```
C100 ORG #C100
C100 20 14 START0 BRA START1
C102 01 2E 01 3A FCB #01,02E,001,03A,001
C106 01 FCC / RENAMED BY JR/
C107 20 52 45 46
C10B 41 53 48 45
C10F 44 20 42 59
C113 20 4A 52
C116 86 00 START1 LDA #900 01 FOR SYS DRV AS DEFAULT
```

```
C118 B7 CC0D
C11B 30 8D 016D
C11F BD CD2D
C122 EC 04
C124 10B3 AFA6
C128 26 10
C12A EC 06
C12C 10B3 0B00
C130 26 08
C132 8C 08
C13A E3 0A
C136 1027 00C2
C13A CC 4CAF LOGST
C13D ED 0C
C13F 86 47
C141 ED 0E
C143 FC CC43
C146 ED 8D 00FA
C14A FC CD13
C14D ED 8D 00F3
C151 FC CD0D
C15A ED 8D 00FD
C15B FC 0A 4
C15D ED 8D 00E8
C15F FC CD2B
C162 ED 8D 00DC
C166 03 01B
C169 25 62
C16B ED 8D 00 1
C16F 1F 03
C171 33 41
C173 CC 0047
C176 31 8D 00C4
C17A 8D 42
C17C 30 C9 004E
C180 AF C9 000E
C184 86 92
C186 AF 84
C188 8D 0406
C18B 26 2B
C18D EC C9 0002
C191 FD CC2B
C194 30 C9 001F
C198 BF 0404
C19B 30 C9 0000
C19F BF CD13
C1A2 3 C9 003F
C1A6 BF CD0D
C1A9 CC 3E3E
C1AC FD CC4E
C1AF B7 CC50
C1B2 7E CD03
C1B5 8D CD3F
C1B8 8D 0403
C1BB 7E CD03
```

```
STA DEFDRV
LEAX LOGFCB,PCB
JBR GETFIL
LDD 4,X
CMPD #04FA6
BNE LOGST
LDD 6,X
CMPD #04600
BNE LOGST
LDD 8,X
ADD 10,X
LDBO LOGEND
LDD #04C F
STD FCBE1,X
LDA #047
STD 14,X
LDD RETADR
STD RETA,PCB
LDD OUTCH2+1
STD OUTCH,PCB
LDD INCH2+1
STD IVEC,PCB
LDD FMSCLS+1
STD CLBV,PCB
LDD MEMEND
STD MEMO,PCB SAVE OLD MEMEND
SUBD #SIZE
PCB #2LOW
STD OWNEND,PCB SAVE NEW MEMEND
YR 0,U
LEAX 1,U NEW START ADDRESS
LDD #LENGTH/2 NUMBER OF BYTES TO XFER
LEAV BASE,PCB GET OLD START ADDRESS
BLKXFR BLOCK MOVE ROUTINE
LEAX LOGFCB-BASE,U
STX SAVFCB-BASE,U SAVE LOG FCB
LDD 42
STD 0,X
JBR FMS
BNE 3A0
LDD OWNEND-04BE,U
STD MEMEND GET NEW MEMEND
LEAX DCLOSE-BASE,U
STX FMSCLS+1
LEAX BASE-BASE,U
STX OUTCH2+1
LEAX INPUT-BASE,U
STX INCH2+1
LDD #03E3E
STD PROMPT
STA PROMPT+2
JMP WARMS
JBR PSTERR
JBR FMSCLS
JMP WARMS
```

- BLOCK MOVE ROUTINE
- D=LENGTH/2
- Y=OLD START ADDRESS
- U=NEW START ADDRESS

```
C1BE 34 4D BLKXFR PSMB U
C1C0 4C 1MCR
C1C1 AE R1 BL LDX Y--
C1C3 AF C1 BTX Y++
C1C5 5A
C1C6 26 F9 BNE 01
C1C8 4A DECA
C1C9 26 F6 BNE 01
C1CB 35 C0 PULB U,PC
C1CD 30 8D 0005 #2LOW
C1D1 8D CD1E LEAX MSGO,PCB
C1D4 31 8D 0065 JBR PSTARR
C1D8 20 37 BRR RESTOR
C1DA 20 2D 20 4E #B04 FCC /-- NOT ENOUGH MEMORY FOR A LOG --/,004
C1DE AF 34 20 45
C1E2 AE AF 55 47
C1E6 48 20 4D 45
C1EA 4D AF 52 59
C1EE 20 46 AF 52
C1F2 20 41 20 4C
C1F6 AF 47 20 2D
C1FA 20 04
C1FC 10BE CD13 LOGEND
C200 AE 2E LDX 14,Y
C202 86 04 LDA #4
C204 A7 84 STA 0,X
C206 8D 0406 JBR FMS
C209 27 06 BEO RESTOR
C20B 8D CD3F JBR PSTERR
C20E 7E CD03 JMP WARMS
C211 EC 22 LDD 2,Y
C213 10B3 CC2B CMDD MEMEND
C 17 26 05 FNE NONECH
C219 EC 24 LDD #,Y
C21B FD CC2B STD MEMO
C21E EC 26 LDD 6,Y
C220 FD CC43 STD RETADR
C223 EC 28 LDD 8,Y
C225 FD CD13 STD OUTCH2+1
C228 EC 2A LDD 10,Y
C22A FD CD0D STD INCH2+1
C22D EC 2C LDD 12,Y
C22F FD D404 STD FMSCLS+1
C232 EC 2B2B LDD #03E3E
C235 FD CC4E STD PROMPT
C238 B7 CD50 STA PROMPT+2
C23B 7E CD03 JMP WARMS
```

• THE CODE WHICH FOLLOWS IS RELOCATED

```
C23E 20 0E BASE BRA OUTP1
C240 OMEND RMB 2
C242 MEMD RMB 2
C244 RETA RMB 1
C245 RETB RMB 1
C246 OVEC RMB 2
C248 IVEC RMB 2
```

CHECK TO SEE IF ANOTHER PROGRAM ALTERED MEMEND. GET ORIGINAL MEMEND

```

C24A      CLSV      AMB      2
C24C      SAVFCB      AMB      2
C24E 3A      16      OUTPT1 PSMB      A,B,X
C250 30      8C 39      LEAX      (LOGFCB,PCB)
C253 8D      D406      JSR      FMS
C256 30      8C ED      LEAX      (OVEC,PCR)
C259 AD      9A      JSR      (O,X)
C25B 35      96      PULS      A,B,X,PC
C25D 34      16      PSMB      A,B,X
C25F 0E      D409      LDX      FCBPTA
C262 27      F7      BEO      OUTRET
C264 EC      8A      CLLIST      LDD      O,X
C266 34      06      PSMB      A,B
C268 30      88 EA      LEAX      -28,X
C26B AC      8C DE      CMPX      (SAVFCB,PCB)
C26E 27      07      BEO      (IGNORE)
C270 86      04      LDA      0A
C272 A7      84      STA      O,X
C274 BD      D406      JSR      FMS
C277 AE      E1      IGNORE      LDX      O,5++
C279 26      E9      BNE      CLLIST
C27B 35      96      PULB      A,B,X,PC
C27D 34      14      PSMB      B,X
C27F 30      8C C6      LEAX      (IVEC,PCB)
C282 AD      9A      JSR      (O,X)
C284 30      8C D5      LEAX      (LOGFCB,PCB)
C287 BD      D406      JSR      FMS
C28A 35      94      PULS      B,X,PC

C28C      C28C      LOGFCB      EQU      *
C28C      C2CC      OUTBUF      EQU      *40
C2CC      C2CC      OUTBUF      EQU      *
C2CC      C2CC      OUTBUF      EQU      $100

O08E      LENGTH      EQU      OUTBUF-BASE
O18E      SIZE      EQU      *-BASE

END      STARTO

```

0 ERROR(S) DETECTED

ESP Electronic Specialists, Inc.

171 South Main Street, Natick, Mass. 01760

(617) 855-1532

SOFTWARE PROTECTION GUIDE & CATALOG

Electronic Specialists is offering a 40 page color

catalog describing power line problems such as noise

and high voltage spikes. Damaging and disruptive

effects on Program execution are described.

Typical Software problems and suggested solutions

are included. Hundreds of protective and interference

cure products are described. Request Catalog 851.

Don Williams, Sr.
68' Micro Journal
5900 Cassandra Smith
P.O. Box 849
Mixon, TN 37343

David R. Forrest
1834 Woodfin Court
Kirkwood, Missouri 63122
(314) 965-6577

Dear Don:

Enclosed find a disk with a utility I wrote that I find very useful. I program primarily in BASIC and have need of hex conversions quite often. Perhaps others would also find it useful.

I've been reading the Journal since its inception and use something from almost every issue. Thanks!

Sincerely yours,

David R. Forrest
David Forrest

* HEX CONVERSION UTILITY

* PLACED IN PUBLIC DOMAIN 10 JAN 85

* D. R. FORREST
* 1834 WOODFIN COURT
* KIRKWOOD, MISSOURI 63122
* (314) 965-6577

* POSITION INDEPENDENT CODE
* LOADS IN UTILITY COMMAND SPACE (\$C100)

* SYNTAX OF COMMAND ARGUMENT CAN BE HEX(1) OR DECIMAL

* ++HEX (ARGUMENT)

* EXAMPLES HEX \$C003 or HEX 256

* DOS EQUATES

```

C003  WARRS      EQU      $C003      DOS WARMSTART
C018  PUTCHR      EQU      $C018      PUT CHARACTER ROUTINE
C024  PCRLF      EQU      $C024      CARRIAGE RETURN LINE FEED
C042  GETHEX      EQU      $C042      GET HEX NUMBER FROM BUFFER
C048  INDEC      EQU      $C048      GET DECIMAL NUMBER FROM BUFFER
C039  OUTDEC      EQU      $C039      OUTPUT DECIMAL NUMBER
C03C  OUTHEX      EQU      $C03C      OUTPUT HEX NUMBER
C027  NITX      EQU      $C027      GET NEXT BUFFER CHARACTER
C015  NITPNT      EQU      $C015      POINTER TO NEXT BUFFER CHARACTER
C01E  PSTANG      EQU      $C01E      PRINT STRINGS $4 TERMINATES

```

```

C100      ORG      $C100
C100 20 03      HEX      BRA      LIST      RUNS IN UTILITY COMMAND SPACE

```

```

C102 02      VN      FCB      2      VERSION NUMBER
C103 0002      HEXND      FCB      2      HEX NUMBER

```

```

C105 8D C027      LIST      JSR      NITX      GET NEXT CHARACTER
C108 81 24      CMA      0's      SEE IF HEX SIGN
C10A 26 05      BNE      DEC      CHECK DECIMAL
C10C 8D C042      JSR      GETHEX      GET HEX NUMBER
C10F 20 00      BRA      OUTPUT      GO PRINT IT

```

* IF NOT HEX SIGN

```

C111 81 20      BEC      CMA      $620      SKIP
C113 27 F0      BEQ      LIST      SPACES
C115 7A C015      BEC      NITPNT      BACK UP POINTER
C118 8D C048      JSR      INDEC      GET DECIMAL NUMBER
C118 12      NOP      DASP INTO OUTPUT

```

* OUTPUT ROUTINE

```

C11C 25 35      OUTPUT      BCS      ERROR      IF ERROR
C11E AF 8C E2      STX      HEXND,PCR      STORE BINARY NUMBER
C121 30 8C DF      LEAX      HEXND,PCR      POINT TO NUMBER
C124 8D C024      JSR      PCRLF      NEW LINE TO PRINT ON
C127 86 24      LDA      0's      SET UP HEX SYMBOL
C129 8D C018      JSR      PUTCHR      OUTPUT HEX SIGN
C12C 8D C03C      JSR      OUTHEX      OUTPUT HEX HIGH ORDER
C12F 30 01      LEAX      1,1      STEP TO LOW ORDER
C131 8D C03C      JSR      OUTHEX      OUTPUT HEX LOW ORDER
C134 30 8C CC      LEAX      HEXND,PCR      POINT TO NUMBER AGAIN
C137 86 20      LDA      $620      OUTPUT A
C139 8D C018      JSR      PUTCHR      SPACE
C13C 8D C039      JSR      OUTDEC      OUTPUT DECIMAL NUMBER
C13F 8D C024      JSR      PCRLF      SPACE TO NEW LINE
C142 7E C003      JMP      WARRS      RETURN TO DOS

```

* ERRORS*

```

C145 49 4E 56 41      INVAL      FCC      'INVALID INPUT', $04
C149 4C 49 44 20
C14D 49 4E 50 55
C151 54 04

```

```

C153 30 8C EF      ERROR      LEAX      INVAL,PCR      POINT TO MESSAGE
C156 8D C01E      JSR      PSTANG
C159 8D C024      JSR      PCRLF      CLEAR BUFFER IF REDIRECTED PRINTER
C15C 7E C003      JMP      WARRS      AND QUIT
C15C 7E C003      END      HEX

```

0 ERROR(S) DETECTED

SYMBOL TABLE:

```

DEC      C111      ERROR      C153      GETHEX      C042      HEX      C100      HEXND      C103
INDEC      C048      INVAL      C145      LIST      C105      NITX      C027      NITPNT      C015
OUTDEC      C039      OUTHEX      C03C      OUTPUT      C11C      PCRLF      C024      PSTANG      C01E
PUTCHR      C018      VN      C102      WARRS      C003

```

Classified Advertising

TELETYPE Model 43 PRINTER - with serial (RS232) interface, and full ASCII keyboard. **LIKE NEW** - New cost \$1295.00-ONLY **\$759.00** ready to run.

Call Tom - Bob, CPI (615) 842-4600

For Sale: Motorola 128K Memory Boards, removed from SWPTC S/O9 \$795.00, SWPTC 8212 Terminals Demonstrators \$795.00, Hazelwood Dynamic 64K Memory Boards \$395.00 Call ask for Tom (615) 842-4600

Wanted to Buy: Used 6800 & 6809 FLEX software. Commercial and public domain. Send description and asking price to: John Current, P.O. Box 273 Honolulu, HI 96859

OS-9™ SOFTWARE

SDISK—Standard disk driver module allows the use of 35, 40, or 80 track double sided drives with COCO OS-9 plus you can read/write/format the OS-9 formats used by other OS-9 systems. **\$29.95**

SDISK + BOOTFIX—As above plus boot directly from a double sided diskette **\$35.95**

FILTER KIT #1—Eleven OS-9 utilities for "wild card" directory lists, copies, moves, deletes, sorts, etc. Now includes disk sector edit utility also. **\$29.95 (\$31.95)**

FILTER KIT #2—Macgen command macro generator builds new commands by combining old ones with parameter substitution, 10 other utilities. **\$29.95 (\$31.95)**

HACKER'S KIT #1—Disassembler and related utilities allow disassembly from memory, file. **\$24.95 (\$26.95)**

PC-XFER UTILITIES —Utilities to read/write and format MS-DOS™ diskettes on CoCo under OS-9. Also transfer files between RS disk basic and OS-9 (requires sdisk). **\$45.00**

SS-50 USERS: Half price closeout of 256K dynamic ram boards, making way for new Megabyte design.

BOLD prices are CoCo OS-9 format disk, other formats (in parenthesis) specify format and OS-9 level. All orders prepaid or COD, VISA and MasterCard accepted. Add \$1.50 S&H on prepaid, COD actual charges added.

**D.P. Johnson, 7655 S.W. Cedarcrest St.
Portland, OR 97223 (503) 244-8152**
(For best service call between 9:11 AM Pacific Time.)

OS-9 is a trademark of Microware and Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.

Selling GIMIX 6809 computer with clock, DMA disk controller, 56K RAM. FLEX-9, OS-9 software. Only \$1295.
John Pomeroy (216) 372-4457.

6809 COMPUTER SYSTEM: SWTP mother board, & Power supply, Percom CPU with serial, serial board, 48K RAM, dual 5inch drives, SD Floppy controller. BEST OFFER! Perkin Elmer 550 CRT terminal, Soroc IQ 120 CRT terminal, two TEK-COM acoustic mode (300). Joe Wicklund (206) 481-2812 evenings (PST)

WANTED: SWTPc MP-LA, Used or Public Domain FLEX 6800 Software. Any help appreciated. Call Craig after 5:00 EST (518) 439-6770

SWTPC S/O9 Computer
64K, CT-82 Terminal, Dual Floppies, 4 Serial Ports, 1 Parallel Port, S-32 Card, FLEX 9.0, TSC X BASIC, Editor, Assembler, Text Processor, Pascal, Debugger, Utilities, More. Excellent Condition.
Keith Teague (405) 624-5158 Weekdays, (405) 624-0621 Evenings.

COMPILER EVALUATION SERVICES By: Ron Anderson

The S.E. MEDIA Division of Computer
Publishing Inc.
Is offering the following "SUBSCRIBER
SERVICE":

COMPILER COMPARISON AND EVALUATION REPORT

Due to the constant and rapid updating and enhancement of numerous compilers, and the different utility, appeal, speed, level of communication, memory usage, etc., of different compilers, the following services are now being offered with periodic updates.

This service, with updates, will allow you who are wary or confused by the various claims of compiler vendors, an opportunity to review comparisons, comments, benchmarks, etc., concerning the many different compilers on the market, for the 6809 microcomputer. Thus the savings could far offset the small cost of this service.

Many have purchased compilers and then discovered that the particular compiler purchased either is not the most efficient for their purposes or does not contain features necessary for their application. Thus the added expense of purchasing additional compiler(s) or not being able to fully utilize the advantages of high level language compilers becomes too expensive.

The following COMPILERS are reviewed initially, more will be reviewed, compared and benchmarked as they become available to the author:

PASCAL "C" GSPL WHIMISCAL PL/9

Initial Subscription - \$39.95
(Includes 1 year updates)
Updates for 1 year - \$14.50

S.E. MEDIA - CPI
5900 Cassandra Smith, POB 794
Hixson, TN 37343
615 842-4601

Give Your OS-9 System The Power It Deserves!
Total Management Planning Software presents

The **TMP** POWER SERIES

New!
Uniflex Version!

The **TMP FREEFORM/FILER** The Premier Text Filing Program!

UNIQUE CAPABILITIES:

- ✓ **THE OBJECTIVE:** For those who want to randomly store information, but retrieve it quickly, the **FREEFORM/FILER** is designed to bridge the gap between word processing and traditional database management programs.
- ✓ **THE KEYWORD SYSTEM:** As you enter or edit your text, you can select any word as a "KEYWORD" or "KEYPHRASE." Each "CARD" can contain as many as 117 KEYWORDS, and up to nine pages of text similar to a 3" x 5" card, with no field restrictions. Each "FILE DRAWER" can contain up to 32,767 pages!
- ✓ **THE SEARCH:** Search for the Card Title, KEYWORD, or a combination of both. Plus, "WILD CARD SEARCHES" save time!
- ✓ **THE RESULTS:** List the titles of cards found, print, or write the cards to a disk file for later printing or use with a word processor.

GIVE YOU MORE ABILITIES:

- **BUSINESS:** Appointments, office phone and address indexes, inventory, service calls, vendor lists, and sales orders.
 - **WRITERS AND RESEARCHERS:** For indexing, cross-referencing or cataloging, and legal and medical research.
 - **PERSONAL:** The possibilities are endless... art and coin collections, bill paying, tax records, and home inventory.
- NOTE:** The **FREEFORM/FILER** integrates data into the "POWER MANAGER."
- '68' MICRO JOURNAL Said:** "TMP FREEFORM/FILER is a flexible Program that can be used for a multitude of tasks without requiring a Computer Science background... a very useful Program."
- Requires 128K, OS-9 Version, \$195; Uniflex version, \$295.**

The **TMP POWER MANAGER**

... Best In Its Class!

POWERFUL CAPABILITIES:

- ✓ More characters per record (7600) than any other program in its class!
- ✓ Each database can contain 32,767 records, with up to 150 fields in each record and up to 50 characters in each field.
- ✓ Powerful Sort and Report Generation capabilities.
- ✓ SORTING on any field or on a combination of fields.
- ✓ Intricate math CALCULATING between fields.

GIVE YOU MORE ABILITIES:

- **POWER MANAGER** can create CUSTOMIZED LETTERS, INVOICES, COLUMNAR REPORTS, OR LABEL FORMATS for mailing!
- Our Users put **POWER MANAGER** to work for them to do: ... customer mailing ... past due notices ... invoicing ... sales analysis ... inventories ... credit, insurance and employee records, client profile reports, tracking stock portfolios, and much more.
- ▶ **THE BOTTOM LINE:** The **POWER MANAGER** is the best in its class! Requires 128K, \$385. (OS-9 Only)

The **TMP POWER PLANNER**

... Unequalled in Speed and Power!

- ★ The **POWER PLANNER** is an electronic spreadsheet with extra Speed and Power due to a unique feature called "circular referencing" that recalculates only the related cells in the spreadsheet.
- Data and formulas can be entered IN ANY ORDER. And, you don't have to keep recalculating for the right answer as in other spreadsheets.
- ★ **SPECIALIZED REPORTS** can be easily created by overlaying any number of screens and automatically updating one spreadsheet with another!
- Other features include "Snap-Shot" printing, full 13-digit precision, standard arithmetic and trig functions, and a worksheet that will display up to 264 rows by 266 columns.
- ★ The Speed and Power of the **POWER PLANNER** make it a natural for Budgeting, Cash Flow Comparisons, Sales Forecasts, Profit/Loss Projections, and all kinds of Financial Analysis and "What If" Calculations. Requires 64K, \$250 (OS-9 Only).

The **TMP FRONTEND** ... A Powerful Menu Program

- ★ The **FRONTEND** allows the user to call up TMP or other programs such as application programs, editors, shell scripts, or commands, from a menu system. Capacity is seven menu screens, each with up to 36 menu options, producing 252 options. Can be called from non-TMP programs. Requires 64K, OS-9 Version \$50, Uniflex Version, \$75.

The **TMP LABELER**

- Lets you make large quantities of labels in seconds, with options for automatic numbering and selecting the number of copies of each label. Variable pitch and 8-line screen. Great for Serial Numbers, Inventory, or making a quick Shipping Label. \$75, Uniflex Only.

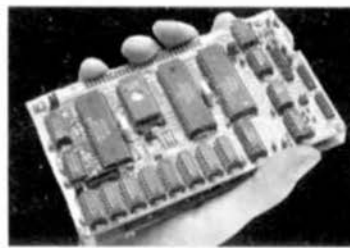
▶ ORDERING INFORMATION: **TMP SOFTWARE**

2431 E. Douglas • Wichita, Kansas • 67211

▶ OR CALL TOLL-FREE: 1-800-255-1382 Ext. 47

We accept VISA, MC, AMEX, money orders and checks.

NOTE: The parent company of TMP Software, The United Software Co., is now the distributor and support organization for TMP Software.



Available
Assembled
and Tested

Compact Flexible 6809 Computer

The new ST-2900 system — a complete 64K small business or hobbyist computer is only one of its many possible configurations. Among its features are:

- Small enough to hold in your hand! (Eurocard size: 3.9" x 6.3")
- Two board "system" for greater versatility than single board computers.
- CPU Board — powerful 6809E processor, 16K or 64K RAM, 2K-8K EPROM, 2 RS232 serial ports with software programmable baud rates, 16 bit counter/timer. Run the CPU board all by itself, or plug your own custom board or our FDC board into the expansion connector.
- FDC Board — double-sided/double-density floppy disk controller with adjustment free digital data separator and write precompensation, 2.8 bit parallel ports, 2-16 bit counter/timers, prototyping area.
- Available as bare PC boards or fully assembled and tested boards. All have solder mask both sides plus all-screened component overlay.

• OS-9 for only \$49?

Well, not quite. But that's all you pay for our OS-9 Converter in Package which lets you use the low cost Radio Shack CoCo version of OS-9 on our ST-2900 system. Save \$131 off the suggested list price of OS-9! **No programming is involved.** Supports CoCo OS-9 and standard OS-9 format disks.

- CPU bare board plus EPROM \$45 FLEX Conversion Package \$29
- FDC bare board \$38 OS-9 Conversion Package \$49
- CPU + FDC board set assembled and tested \$329

• Add \$5 shipping/handling (\$10 overseas). These prices are in U.S. funds. Canadian orders; call or write for price. Terms: money order, certified check, VISA, MasterCard is a trademark of Technical Systems Computers. OS-9 is a trademark of Microware and Microsoft.



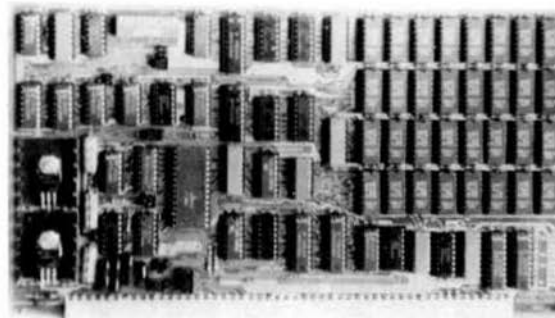
**SARDIS
TECHNOLOGIES**

Write for free brochure and complete price list.

(604) 255-4485 (4-5 pm PST)

2261 E. 11th Ave. Vancouver, B.C., Canada V5N 1Z7

!!! NEW PRICES !!!



256K, 512K, 1 MEG MEMORY SYSTEM

Now compatible with DMA controllers. Runs at up to 2MHz without generating HARDY or interrupts. Has an optional on board DAT for use with CPU cards without a DAT. 128K, 256K, 512K or 1M byte per card. Field upgradeable. Optional configuration allows 4M byte address reach (using memory board DAT) without CPU changes or cables. 1 year limited warranty.

TURBO virtual disk software and memory diagnostics supplied with the system.

Prepaid: 256K:\$695, 128K:\$545, 512K:\$795, 1024K:\$1195

Domestic shipping and handling \$10.00. Users manual: \$15.00, applicable toward system purchase. Cashiers check, COD, or personal checks must clear before shipment. Fls. residents add 5% sales tax. Shipped stock to 30 days. Dealer and gateway discounts available.

COMPUTER EXCELLENCE INC.

P.O. BOX 8442

CORAL SPRINGS, FL 33065

(305) 752-8321

GOOD NEWS!



C **for the** **6809** **WAS NEVER** **BETTER!**

INTROL-C/6809, Version 1.5

Introl's highly acclaimed 6809 C compilers and cross-compilers are now more powerful than ever!

We've incorporated a totally new 6809 Relocating Assembler, Linker and Loader. Initializer support has been added, leaving only bitfield-type structure members and doubles lacking from a 100% full K&R implementation. The Runtime Library has been expanded and the Library Manager is even more versatile and convenient to use. Best of all, compiled code is just as compact and fast-executing as ever - and even a bit more so! A compatible macro assembler, as well as source for the full Runtime Library, are available as extra-cost options.

Resident compilers are available under **Uniflex, Flex** and **OS9**.

Cross-compilers are available for **PDP-11/UNIX** and **IBM PC/PC DOS** hosts.

Trademarks:

Introl-C, Introl Corporation

Flex and Uniflex, Technical Systems Consultants

OS9, Microware Systems

PDP-11, Digital Equipment Corp.

UNIX, Bell Laboratories

IBM PC, International Business Machines

For further information, please call or write.

INTROL
CORPORATION

647 W. Virginia St.
Milwaukee, WI 53204
(414) 276-2937

FEATURES THE
POWERFUL, THIRD
GENERATION,
MOTOROLA 6809
PROCESSOR!

THE 6809 "UNIBOARD"™ SINGLE BOARD COMPUTER KIT

PERFECT FOR COLLEGES, OEM'S, INDUSTRIAL
AND SCIENTIFIC USES!

64K RAM! DOUBLE DENSITY
FLOPPY DISK CONTROLLER!

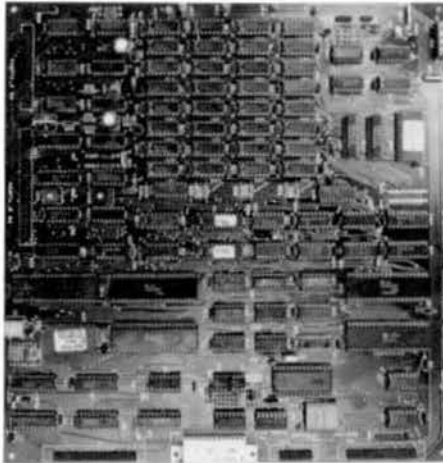
New!
Lower Price!

BLANK PC BOARD

\$99⁹⁵

WITH PAL'S, AND
TWO EPROMS.

FOR 5-1/4 OR 8 INCH
SOURCE DISKETTE
ADD \$10.



\$249⁰⁰

COMPLETE KIT!
FULLY SOCKETED.

**PRICE
CUT!!**

THE COMPACTA UNIBOARD™: Through special arrangement with COMPACTA INC., we are proud to have been selected the exclusive U.S. Mfg. of their new 6809 UNIBOARD™ COMPUTER KIT. Many software professionals feel that the 6809 features probably the most powerful instruction set available today on ANY 8 bit micro. Now, at last, all of that immense computing power is available at a truly unbelievably low price.

FEATURES:

- ★ 64K RAM using 4116 RAMS.
- ★ 6809E Motorola CPU.
- ★ Double Density Floppy Disk Controller for either 5-1/4 or 8 inch drives. Uses WD1793.
- ★ On board 80 x 24 video for a low cost console. Uses 2716 Char. Gen. Programmable Formats. Uses 6845 CRT Controller.
- ★ ASCII keyboard parallel input interface. (6522)
- ★ Serial I/O (6551) for RS232C or 20 MA loop.
- ★ Centronics compatible parallel printer interface. (6522)
- ★ Buss expansion Interface with DMA channel. (6844)
- ★ Dual timer for real time clock application.
- ★ Powerful on board system monitor (2732). Features commands such as Go To, Alter, Fill, Move, Display, or Test Memory. Also Read and Write Sectors. Boot Normal, Unknown, and General Flex™.

YOUR CHOICE OF POPULAR DISK OPERATING SYSTEMS:

FLEX™ from TSC	\$99
OS9™ from Microware	\$199
Specify 5-1/4 or 8 Inch	

PC BOARD IS
DOUBLE SIDED, PLATED THRU
SOLDER MASKED, 11 x 11-1/2 IN.

ALL SALES ARE MADE SUBJECT TO THE TERMS OF OUR 90 DAY
LIMITED WARRANTY. A FREE COPY IS AVAILABLE UPON REQUEST.

Digital Research Computers
(OF TEXAS)

P.O. BOX 461565 • GARLAND, TEXAS 75046 • (214) 225-2309

TERMS: Shipments will be made approximately 3 to 6 weeks after we
receive your order. VISA, MC, cash accepted. Add \$4.00 shipping.
USA AND CANADA ONLY

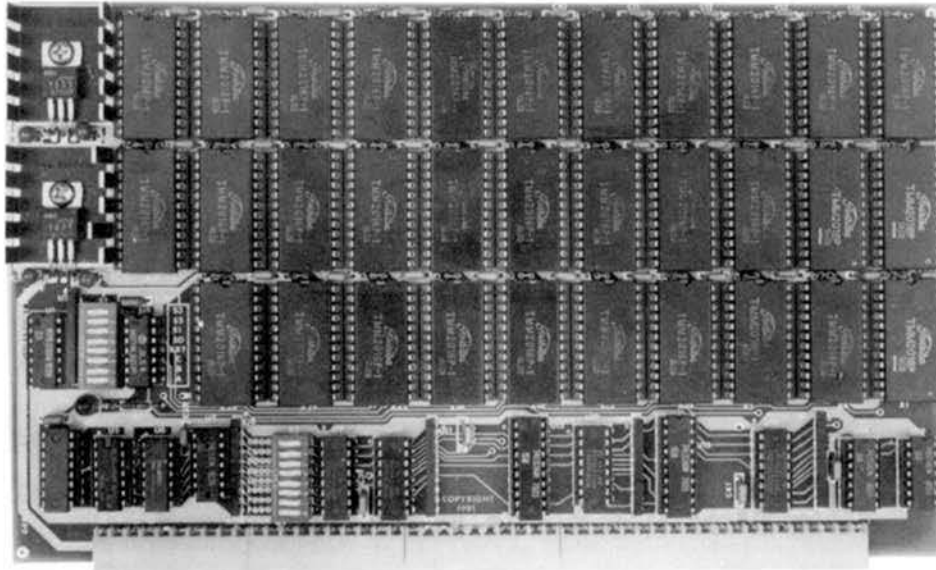
64K SS-50 STATIC RAM

PRICE CUT!!

\$149⁰⁰
(48K KIT)

NEW!

LOW
POWER!



RAM
OR
EPROM!

BLANK PC BOARD
WITH DOCUMENTATION
\$45

SUPPORT ICs + CAPS - \$18.00
FULL SOCKET SET - \$15.00

ASSEMBLED AND TESTED ADD \$50

FEATURES:

- ★ Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
- ★ Fully supports Extended Addressing.
- ★ 64K draws only approximately 500 MA.
- ★ 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
- ★ Board is configured as 3-16K blocks and 8-2K blocks (within any 64K block) for maximum flexibility.
- ★ 2716 EPROMs may be installed anywhere on Board.
- ★ Top 16K may be disabled in 2K blocks to avoid any I/O conflicts.
- ★ One Board supports both RAM and EPROM.
- ★ RAM supports 2MHZ operation at no extra charge!
- ★ Board may be partially populated in 16K increments.

56K	\$169
64K	\$199

16K STATIC RAMS?

The new 2K x 8, 24 PIN, static RAMs are the next generation of high density, high speed, low power, RAMs. Pioneered by such companies as HITACHI and TOSHIBA, and soon to be second sourced by most major U.S. manufacturers, these ultra low power parts, feature 2716 compatible pin out. Thus fully interchangeable ROM/RAM boards are at last a reality, and you get BLINDING speed and LOW power thrown in for virtually nothing.

CLOSE OUT SPECIAL
WE HAVE DROPPED OUR 32K SS-50 STATIC RAM BOARD WHICH USED 2114 LOW POWER RAMS. WE WILL SELL THE REMAINING STOCK OF BLANK PCB'S WITH DATA FOR \$17.50 EA. THESE FORMERLY SOLD FOR \$50.

Digital Research Computers
(OF TEXAS)

P.O. BOX 461565 • GARLAND, TEXAS 75046 • (214) 225-2309

TERMS: Add \$2.00 postage. We pay balance. Order under \$15 add 75c handling. No C.O.D. We accept Visa and MasterCard. Tex. Res. add 5% Tax. Foreign orders (except Canada) add 20% P & H. Orders over \$50, add 85c for insurance.

DISKETTES AND 680X SOFTWARE

SUPER SLEUTH DISASSEMBLER EACH \$99-FLEX, \$101-OS-9, \$100-UNIFLEX

interactively generates source on disk with labels, includes xref, label definition, binary file editing, etc.

specify 6800, 1.2, 3.5, 8/6502 version or Z-80/8080/85 version
OS-9 and UNIFLEX versions also process FLEX object file formats

OBJECT ONLY versions: EACH \$50-FLEX & OS-9, \$49-COCO DOS
COCO DOS available in 6800, 1.2, 3.5, 8/6502 version only

CROSS-ASSEMBLERS EACH \$50-FLEX/UNIFLEX/OS-9, ANY 3 \$100, ALL \$200

specify for 180x, 650x, 680x, Z-80, 8048/51, 8085, 68000
true, modular, free-standing cross-assemblers, written in C
8-bit source included only with all cross-assemblers (for \$200)

DEBUGGING SIMULATORS EACH \$75-FLEX, \$100-OS-9, \$80-UNIFLEX

specify 6800/1, (14)6805, 6502, 6809 OS-9, Z-80 FLEX

OBJECT ONLY versions: EACH \$50-COCO FLEX & COCO OS-9

6502 TO 6809 ASSEMBLER TRANSLATOR \$75-FLEX, \$85-OS-9, \$80-UNIFLEX

translates 6502 programs to 6809, noting inexact conversions

6800 TO 6809 & 6809 PIC TRANSLATORS \$50-FLEX, \$75-OS-9, \$60-UNIFLEX

translates 6800 programs to 6809, 6809 programs to PIC

FULL-SCREEN FLEX AND UNIFLEX TSC XBASIC PROGRAMS FOR 6809

(with complete cursor control)

DISPLAY GENERATOR/DOCUMENTOR

\$50 w/source, \$25 without

MAILING LIST SYSTEM

\$100 w/source, \$50 without

INVENTORY WITH MRP

\$100 w/source, \$50 without

TABULA RASA SPREADSHEET

\$100 w/source, \$50 without

DISK AND XBASIC UTILITY PROGRAM LIBRARY \$50-FLEX & UNIFLEX

edit sectors, sort directory, maintain master catalog, do disk sorts, xref BASIC, ...

CMODEM PROGRAM \$100-FLEX & OS-9 & UNIFLEX, OBJECT-ONLY EACH \$50

provides menu-driven telecommunications facilities, with terminal mode, up/down load, MODEM7 protocol, etc.

5.25" SOFT-SECTORED DISKS EACH 10-PACK \$13-SSSD \$15-SSDD/DSDD \$25-DSQD

American-made, excellent quality, with tickets and hub rings

SS-50C 256K 1.5MHZ MEMORY BOARDS BLANK \$100 A&T \$350

with instruction manual, schematics, and delay line; all parts readily available

Most programs in source on disk; specify computer, disk size, operating system.

Contact CSC for full catalog and dealer information.

25% discount for multiple purchases of same program on same order.

For VISA and MASTER CARD, give account, exp. date, phone. US funds only.

Add GA sales tax and 5% shipping; no shipping for disks in 100's.

(UNIFLEX trademark Technical Systems Consultants, OS-9 trademark Microware.

Computer Systems Consultants, Inc.

1454 Latta Lane, Conyers, GA 30207

Telephone Number 404-483-1717/4570

SOFTWARE FOR THE HARDWARE

** FORTH PROGRAMMING TOOLS from the 68XX&X **
** FORTH specialists — get the best!! **

NOW AVAILABLE — A variety of rom and disk FORTH systems to run on and/or do TARGET COMPILATION for

6800, 6301/6801, 6809, 68000, 8080, Z80

Write or call for information on a special system to fit your requirement.

Standard systems available for these hardware—

EPSON HX-20 rom system and target compiler
6809 rom systems for SS-50, EXORCISER, STD, ETC.
COLOR COMPUTER
6800/6809 FLEX or EXORCISER disk systems.
68000 rom based systems
68000 CP/M-68K disk systems, MODEL II/12/16

tFORTH is a refined version of FORTH Interest Group standard FORTH, faster than FIG-FORTH. FORTH is both a compiler and an interpreter. It executes orders of magnitudes faster than interpretive BASIC. MORE IMPORTANT, CODE DEVELOPMENT AND TESTING is much, much faster than compiled languages such as PASCAL and C. If Software DEVELOPMENT COSTS are an important concern for you, you need tFORTH!

firmFORTH[™] is for the programmer who needs to squeeze the most into roms. It is a professional programmer's tool for compact rommable code for controller applications.

~ tFORTH and firmFORTH are trademarks of Talbot Microsystems.

~ FLEX is a trademark of Technical Systems Consultants, Inc.

~ CP/M-68K is trademark of Digital Research, Inc.

tFORTH[™]
from TALBOT MICROSYSTEMS
NEW SYSTEMS FOR
6301/6801, 6809, and 68000

---> tFORTH SYSTEMS <---

For all FLEX systems: GIMIX, SWTP, SS8, or EXORCISER Specify 5 or 8 inch diskette, hardware type, and 6800 or 6809.

** tFORTH — extended fig FORTH (1 disk) \$100 (\$15)
with fig line editor.

** tFORTH+ — more! (3 5" or 2 8" disks) \$250 (\$25)
adds screen editor, assembler, extended data types, utilities, games, and debugging aids.

** TRS-80 COLORFORTH — available from The Micro Works

** firm FORTH — 6809 only \$350 (\$10)
For target compilations to rommable code.
Automatically deletes unused code. Includes HOST system source and target nucleus source. No royalty on targets. Requires but does not include tFORTH+.

** FORTH PROGRAMMING AIDS — elaborate decompiler \$150

** tFORTH for HX-20, in 16K roms for expansion unit or replace BASIC \$170

** tFORTH-68K for CP/M-68K 8" disk system \$290
Makes Model 16 a super software development system.

** Nautilus Systems Cross Compiler

— Requires: tFORTH + HOST + at least one TARGET;
— HOST system code (6809 or 68000) \$200

— TARGET source code: 6800-\$200, 6301/6801—\$200
same plus HX-20 extensions— \$300

6809—\$300, 8080/Z80—\$200, 68000—\$350

Manuals available separately — price in ().
Add \$6/system for shipping, \$15 for foreign air.

TALBOT MICROSYSTEMS 1927 Curtis Ave., Redondo Beach, CA 90278 (213) 376 9941

WINDRUSH MICRO SYSTEMS

UPROM II



PROGRAMS and VERIFIES: 12750, 12500, 12710, 12510, 12732/2732A, MEM00704/0, 12704/2704A, 12504, 127120/27120A, and 127250. (Intel, TMS, and Motorola).

NO PERSONALITY MODULES REQUIRED!

TRI-VOLT EPROMS ARE NOT SUPPORTED

Intel's intelligent programming (ia) implemented for Intel 2704, 27120 and 27250 devices. Intelligent programming reduces the average programming time of a 2704 from 7 minutes to 1 minute 15 seconds (under FLEX) with greatly improved reliability.

fully enclosed pod with 5' of flat ribbon cable for connection to the host computer MC6821 PIA interface board.

MC6809 software for FLEX and OS9 (Level 1 or 2, Version 1.2).

BINARY DISK FILE offset loader supplied with FLEX, ADOS and OS9.

Menu driven software provides the following facilities:

- a. FILL a selected area of the buffer with a HEX char.
- b. MOVE blocks of data.
- c. DUMP the buffer in HEX and ASCII.
- d. FIND a string of bytes in the buffer.
- e. EXAMINE/CHANGE the contents of the buffer.
- f. CAC checksum a selected area of the buffer.
- g. COPY a selected area of an EPROM into the buffer.
- h. VERIFY a selected area of an EPROM against the buffer.
- i. PROGRAM a selected area of an EPROM with data in the buffer.
- j. SELECT a new EPROM type (return to types menu).
- k. ENTER to the system monitor.
- l. RETURN to the operating system.
- m. EXECUTE any DOS utility (only in FLEX and OS9 versions).

FLEX AND OS9 VERSIONS AVAILABLE FROM GEMIX. SSO/ADOS CONTACT US DIRECT.

PL/9

- Friendly inter-active environment where you have INSTANT access to the Editor, the Compiler, and the Trace-Debugger, which, amongst other things, can single step the program a SOURCE line at a time. You also have direct access to any FLEX utility and your system monitor.
- 375+ page manual organized as a tutorial with plenty of examples.
- Fast SINGLE PASS compiler produces 8K of COMPACT and FAST 6809 machine code output per minute with no run-time overheads or license fees.
- Fully compatible with TSC text editor format disk files.
- Signed and unsigned BYTES and INTEGERS, 32-bit floating point REALs.
- Vectors (single dimension arrays) and pointers are supported.
- Mathematical expressions: (+), (-), (*), (/), modulus (%), negation (~)
- Expression evaluators: (=), (<), (>), (<=), (>=), (=)
- Bit operators: (AND), (OR), (EOR/XOR), (NOT), (SHIFT), (SWAP)
- Logical operators: (AND), (OR), (EOR/XOR)
- Control statements: IF...THEN...ELSE, IF...CASE1...CASE2...ELSE, BEGIN...END, WHILE...REPEAT...UNTIL, REPEAT...FOREVER, CALL, JUMP, RETURN, BREAK, GOTO.
- Direct access to (ACCA), (ACCB), (ACCD), (XREG), (CCR) and (STAK).
- FULLY supports the MC6809 RESET, HALT, FIRO, IRQ, SWI, SWIZ, and SWIS vectors. Writing a self-starting (from power-up) program that uses ANY, or ALL, of the MC6809 interrupts is an absolute snap!
- Machine code may be embedded in the program via the 'GEN' statement. This enables you to code critical routines in assembly language and embed them in the PL/9 program (see 'MACE' for details).
- Procedures may be passed and may return variables. This makes them functions which behave as though they were an integral part of PL/9.
- Several fully documented library procedure modules are supplied: EOSUBS, BITIO, MARIO, NEMIO, FLEXIO, SCIPACK, STRSUBS, BASTRING, and REALCOM.

'... THIS IS THE MOST EFFICIENT COMPILER I HAVE FOUND TO DATE.'

Quoted from Ron Anderson's FLEX User Notes column in '68. Need we say more?

MACE/XMACE/ASM05

All of these products feature a highly productive environment where the editor and the assembler reside in every together. Some are the days of tedious disk load and save operations while you are debugging your code.

- Friendly inter-active environment where you have instant access to the Editor and the Assembler, FLEX utilities and your system monitor.
- MACE can also produce ASMPP05 (GEN statements) for PL/9 with the assembly language source passed to the output as comments.
- XMACE is a cross assembler for the 6800/12/3/8 and supports the extended mnemonics of the 6303.
- ASM05 is a cross assembler for the 6805.

D-BUG

LOOKING for a single step tracer and mini in-line disassembler that is easy to use? Look no further, you have found it. This package is ideal for those small assembly language program debugging sessions. D-BUG occupies less than 6K (including its stack and variables) and may be loaded anywhere in memory. All you do is LOAD IT, AIM IT and GO! (80 col VDU only).

McCOSH 'C'

This is as complete a 'C' compiler as you will find on any operating system for the 6809. It is completely compatible with UNIX V11 and only lacks 'bit-fields' (which are of little practical use in an 8-bit world!).

- Produces very efficient assembly language source output with the 'C' source optionally interleaved as comments.
- Built-in optimizer will shorten object code by about 11%.
- Supports interleaved assembly language programs.
- INCLUDES its own assembler. The TSC relocating assembler is only required if you want to generate your own libraries.
- The pre-processor, compiler, optimizer, assembler and loader all run independently or under the 'CC' executive. 'CC' makes compiling a program to executable object as simple as typing in 'CC,HELLO.C <RETURN>'.

IEEE - 488

- SUPPORTS ALL PRINCIPAL MODES OF THE IEEE-488 (1975/8) BUS SPECIFICATION:
 - Talker
 - Listener
 - System Controller
 - Serial Poll
 - Parallel Poll
 - Group Trigger
 - Single or Dual Primary Address
 - Secondary Address
 - Talk only ... Listen only
- Fully documented with a complete reprint of the KILBAUD article on the IEEE bus and the Motorola publication 'Getting aboard the IEEE Bus'.
- Low level assembly language drivers suitable for 6800, 6801, 6802, 6803, 6808 and 6809 are supplied in the form of listings. A complete back to back test program is also supplied in the form of a listing. These drivers have been extensively tested and are GUARANTEED to work.
- Single 5-30 board (4, 8 or 16 addresses per port), fully socketed, gold plated bus connectors and IEEE interface cable assembly.

PRICES

D-BUG	(6809 FLEX only)	\$ 75.00
MACE	(6809 FLEX only)	\$ 75.00
XMACE	(6809 FLEX only)	\$ 98.00
ASM05	(6809 FLEX only)	\$ 98.00
PL/9	(6809 FLEX only)	\$198.00
'C'	(6809 FLEX only)	\$295.00
IEEE-488	with IEEE-488 cable assembly	\$298.00
UPROM-II/U	with one version of software (no cable or interface) ..	\$395.00
UPROM-II/C	as above but complete with cable and 5-30 interface ..	\$545.00
CABLE	5' multi-pin 50 way cable with 100 connectors	\$ 35.00
5-30 I/O	55-30 interface for UPROM-II	\$130.00
EXOR INT	Motorola EXORbus (EXORciser) interface for UPROM-II ...	\$195.00
UPROM 3FT	Software drivers for 2nd operating system.	
	Specify FLEX or OS9 AND disk size!	\$ 35.00
UPROM SRC	Assembly language source (contact us direct)	

ALL PRICES INCLUDE AIR MAIL POSTAGE

Terms: CWO. Payment by Int'l Money Order, VISA or MASTER-CARD also accepted.

WORSTEAD LABORATORIES, NORTH WALSHAM, NORFOLK, ENGLAND. NR28 9SA.

TEL: 44 (892) 404086
TLX: 975548 WMICRO G

WE STOCK THE FOLLOWING COMPANIES PRODUCTS:
GEMIX, SSI, FHL, MICROWARE, TSC, LUCIDATA, LLOYD I/O, & ALPAC & ASSOCIATES.

FLEX (tm) is a trademark of Technical Systems Consultants, OS-9 (tm) is a trademark of Microware Systems Corporation, MDS (tm) and EXORciser (tm) are trademarks of Motorola Incorporated.

DYNACALC[®]

ELECTRONIC SPREAD-SHEET

NOW FOR **68000**

68000 DYNACALC does everything
6809 DYNACALC does, and more:

Worksheets up to 18278 columns or 9999 rows.
Built-in financial formulas.

Smart terminal support for faster scrolling.
Copy, Blank, Hide, and Format COLUMNS or BLOCKS.
Many new display formats — up to four windows.
More efficient data storage and even greater speed.

Uses existing DYNACALC worksheets.

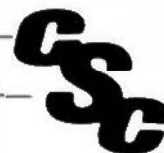
System requirements:

68000-family CPU, OS-9 68k version 1.1 or later.
128k RAM minimum, more preferred.
One or more CRT terminals with cursor addressing.
Printer optional.

Price \$595.00 per single copy;
dealer and OEM inquiries invited.

COMPUTER SYSTEMS CENTER

42 Four Seasons Center #122
Chesterfield, MO 63017 USA
(314) 576-5020



'68' MICRO JOURNAL

- ★ The only ALL 6800 Computer Magazine.
- ★ More 6800 material than all the others combined: **MAGAZINE COMPARISON**

(2 years)

Monthly Averages

KB	BYTE	6800 Articles CC	DOBB'S	TOTAL PAGES
7.8	6.4	2.7	2.2	19.1 ea. mo.

Average cost for all four each month: \$6.53

(Based on advertised 1-year subscription price)

'68' cost per month: \$2.04

That's Right! Much, Much More

for About

1/3 the Cost!

OK, PLEASE ENTER MY SUBSCRIPTION

Bill My: Master Charge ☐ — VISA ☐

Card # _____ Exp. Date _____

For ☐ 1-Year ☐ 2 Years ☐ 3 Years

Enclosed: \$ _____

Name _____

Street _____

City _____ State _____ Zip _____

My Computer Is: _____

68 Micro Journal
5900 Cassandra Smith Rd.
Hixson, TN 37343

Subscription Rates
(Effective March 3, 1985)

U.S.A.: 1 Year \$24.50, 2 Years \$42.50, 3 Years \$64.50

* Foreign Surface: Add \$12.00 per Year to USA Price.

* Foreign Airmail: Add \$48.00 per Year to USA Price.

* Canada & Mexico: Add \$ 9.50 per Year to USA Price.

* U.S. Currency Cash or Check Drawn on a USA Bank ☒



STAR-DOS LEVEL I

Whenever a new DOS is introduced, there's always the problem of developing software to work with it. So we did it the opposite way — we analyzed the requirements of software that already exists and developed a DOS that met them... and exceeded them! The result is STAR-DOS Level I, a new DOS for 6809 systems, ideal for single-user industrial, control, and advanced hobbyist applications. This includes SS-50 systems and single-board computers from a variety of vendors.

Level I is compatible with most current 6809 hardware and software. On the hardware side, it allows up to ten floppy or Winchester drives with appropriate controllers. On the software side, it runs existing 6809 software from all the major 6809 software suppliers, including TSC, Star-Kits, Introl, and others.

Write or call for more information. STAR-KITS Software Systems Corporation, P.O. Box 209, Mt. Kisco N.Y. 10549 (914) 241-0287.



ANDERSON COMPUTER CONSULTANTS & Associates

Ron Anderson, respected author and columnist for 68 MICRO JOURNAL announces the **Anderson Computer Consultants & Associates**, a consulting firm dealing primarily in 68XX(X) software design. Our wide experience in designing 6809 based control systems for machine tools is now available on a consultation basis.

Our experience includes programming machine control functions, signal analysis, multi-axis servo control (CNC) and general software design and development. We have extensive experience in instrumentation and analysis of specialized software. We support all popular languages pertaining to the 6809 and other 68XX(X) processors.

If you are a manufacturer of a control or measuring package that you believe could benefit from efficient software, write or call Ron Anderson. The fact that any calculation you can do with pencil and paper, can be done much better with a microcomputer. We will be happy to review your problem and offer a modern, state-of-the-art microcomputer solution. We can do the entire job or work with your software or hardware engineers.

Anderson Computer Consultants & Associates
3540 Sturbridge Court
Ann Arbor, MI 48105

COMPARE

our EPROM PROGRAMMER with the field.

All data taken directly from manufacturer's current advertising. Software, interfaces, or personality modules may also be required at additional cost.

- Triple voltage EPROM
- Supplied in kit form

		A	B	C	D	E	F
INTERFACE	S30	PAR	PAR	SER	S30	SER	SER
INTELLIGENT	NO	NO	NO	YES	NO	YES	YES
PROGRAMS							
2704*			•				•
2508	•			•	•	•	•
2708*			•				•
2758	•	•	•	•	•	•	•
2516	•		•		•	•	•
2716	•	•	•	•	•	•	•
2716*			•				•
2532	•		•	•	•	•	•
2732	•		•	•	•	•	•
2732A	•		•	•	•	•	•
2584	•		•		•	•	•
2784	•		•		•	•	•
2528	•				•		
27128	•						
2816							•
68784			•				
8748						•	
8749						•	
TOTAL	11	3	12	8	11	11	11
PRICE	\$125	\$45*	\$169	\$289	\$375	\$489	\$575

UNITEK EPROM Programmer, \$125. Personality module for 2508, 2758, 2516, and 2716 included. Specify CPU, disk also, and operating system (TSC's FLEX or SSB's UOE) when ordering. Manual only. \$10: refundable with EPROM purchase.

UNITEK • P.O. Box 671 • Emporia, VA 23847

K-BASIC for OS9 & FLEX \$199

K-BASIC is a complete BASIC compiler package including: the compiler itself; the assembler; documentation; and sample programs. It features six atomic data types including: real numbers; strings; 8 bit, 16 bit, 32 bit, and 64 bit signed integers. All types may be dimensioned with one or two subscripts. K-BASIC converts programs to MACHINE language code which may be put into EPROMS or ROMS.

K-BASIC syntax is very close to TSC's BASIC and XBASIC interpreters. Line numbers are not required (may be up to 16 characters). Variable names may be up to 12 characters long. The AT statement dimensions variables to absolute memory addresses.

The future of K-BASIC will see additional versions for the assorted interpreters currently available. This means you can compile your BASIC programs you now have.

Call (503) 666-1097 for our CATALOG, we have many other programs including: DO...\$69 OSM...\$99 ED/ASM...\$69

CRASMB for OS9 & FLEX \$399

CRASMB is the highly acclaimed cross assembler package for OS9 and FLEX systems, and is the only one of its type available. It turns your computer into a development station for these CPUs:

6800 6801 6804 6805 6809 6811 6502
7000 1802 8048 8051 8080 8085 280
(68000 16/32 bit cross assembler...\$249)

CRASMB features include: Macros, Conditional assembly, Library file calls (12 deep), Symbol length to 30 characters, Symbol cross reference tables, Object code in 4 formats (OS9, FLEX, S1-S9, INTEL HEX), plus many other extended directives and options not found on other assemblers.

LLOYD I/O 19535 NE GLISAN, PORTLAND, OR 97230 USA
Phone: (503) 666-1097 (Software Consultation Available)

VISA, MC, CDD, CHECK, APPROVED P.O.'s ACCEPTED

England: Vivaway (0582 423425), Windrush (0692 405189)

Germany: Zacher Computer (65 25 299)

Australia: Paris Radio Electronics (61 2 344 9111)

OS9 is a ® of Microsoft, FLEX is a ® of TSC

68' MICRO JOURNAL

Disk- 1 Filesort, Minicat, Minicopy, Minifms, **Lifetime, **Poetry, **Foodlist, **Diet.

Disk- 2 Diskedit w/ inst.& fixes, Prime, *Prmod, **Snoopy, **Football, **Hexpaw, **Lifetime

Disk- 3 Cbug09, Sec1, Sec2, Find, Table2, Intext, Disk-exp, *Disksave.

Disk- 4 Mailing Program, *Finddat, *Change, *Testdisk.

DISK- 5 *DISKFIX 1, *DISKFIX 2, **LETTER, **LOVESIGN, **BLACKJAK, **BOWLING.

Disk- 6 **Purchase Order, Index (Disk file indx)

Disk- 7 Linking Loader, Rload, Harkness

Disk- 8 Crtest, Lanpher (May 82)

Disk- 9 Datecopy, Diskfix9 (Aug 82)

Disk-10 Home Accounting (July 82)

Disk-11 Dissembler (June 84)

Disk-12 Modem68 (May 84)

Disk-13 *Initmf68, Testmf68, *Cleanup, *Dskalign, Help

Disk-14 *Init, *Test, *Terminal, *Find, *Diskedit, Init.Lib

Disk-15 Modem9 + Updates (Dec. 84 Gilchrist) to Modem9 (April 84 Commo)

Disk-16 Copy.Txt, Copy.Doc, Cat.Txt, Cat.Doc, Date.Txt

NOTE:

This is a reader service **ONLY!** No Warranty is offered or implied, they are as received by '68' Micro Journal, and are for reader convenience **ONLY** (some MAY include fixes or patches). Also 6800 and 6809 programs are mixed, as each is fairly simple (mostly) to convert to the other.

PRICE: 8" Disk \$29.95 - 5" Disk \$24.95

68 MICRO JOURNAL

POB 794

Hixson, TN 37343

615-842-4600 In Tenn.

1-800-338 6800 In U.S.

* Indicates 6800

** Indicates BASIC SWTPC or TSC
6809 no Indicator.

MASTER CARD - VISA Accepted
Foreign -- add 10% for Surface
or 20% for Air!!

TRS-80 + MOD I, III, COCO, T199/4a
TIMEX 1000, OSBORE, others

GOLD PLUG - 80

Eliminate disk reboots and data loss due to oxidized contacts at the card edge connectors. **GOLD PLUG 80** solders to the board edge connector. Use your existing cables. (if gold plated)

GOLD PLUG 80 Mod I (6)	\$44.95	\$54.95
Keyboard/EI (mod I)	15.95	18.95
Individual connectors	7.95	9.95
COCO Disk Module (2)	16.95	18.95
Ground tab extensions	INCL	1.00
Disk Drives (all R.S.)	7.95	9.95
Gold Disk Cable 2 Drive		29.95
Four Drive Cable		39.95
GOLD PLUG 80 Mod III (6)		54.95
Internal 2 Drive Cable		29.95
Mod III Expansion port		10.95
USA shipping \$1.45		Can/Mex \$4.
Foreign \$7.		TEXAS 5% TAX

SPECIAL
PRICES

COCO MODULE
INSTALLATION
AVAILABLE

Ask your favorite dealer or order direct



E.A.P. CO.
P.O. BOX 14



ORDER TODAY!

KELLER, TEXAS 76248

(817) 498-4242

MC/VISA

+ trademark Tandy Corp

ARCADE 50

POWERFUL COLOR GRAPHICS

Uses the new TMS9918A Video Display processor. High resolution 256 x 192 pixel display with 15 colors. 16K Bytes of onboard RAM does not reduce user memory. 32 graphic images can be individually moved with simple X-Y commands for smooth animation. External Video input allows subtitling. NTSC composite video output.

- SOUND EFFECTS AND MUSIC**
- Three AY3-8910 Programmable Sound Generators
 - Nine simultaneous voices
 - Three independent noise sources
 - Onboard stereo amplifier drives two 8 ohm speakers

ADDITIONAL I/O CAPABILITIES

- Eight analog inputs with 8 bit resolution
- Supports four joysticks with pushbutton switches
- Eight bit parallel I/O port
- Enable unit maps into 256 bytes of memory

FBASIC

TERMINUS DESIGN INC. in conjunction with Microware Systems Corporation is proud to announce FBASIC an enhancement of Microware's 6800/ BASIC. Their fast compiled BASIC has been adopted for 6809 users with added video and sound features for ARCADE 50 users. FBASIC is a true compiler that produces optimized machine language modules which are ROMable and require no Run-Time package. FBASIC requires less memory overhead and runs hundreds of times faster than BASIC interpreters. It supports standard BASIC instruction including string functions. Disk I/O and fast integer arithmetic with multiple precision capability. Graphics verbs and functions fully support the Arcade 50.

ARCADE 50 assembled and tested	\$325.00
Video and Audio connector set	16.00
4 Joystick connector set	15.00
2 Radio Shack Joysticks	24.00
Gold Molex connectors	12.00
FBASIC for 6809	110.00
FBASIC (with ARCADE 50)	110.00
ARCADE 50 RGB	75.00
LABVIDEO (Motorola EXORbus)	375.00
NEW MV09 6809 Processor Board	375.00
256K Dynamic Memory Board	225.00
256K Dynamic Memory Board 1w/64KI	795.00
64K Dynamic Memory Board	395.00
	295.00

TERMINUS DESIGN INC
16 SCARBROUGH ROAD
ELLENWOOD, CA 94049

TERMINUS DESIGN INC. (404) 474-4866

INTELLICOM™

An INTELLIGent COMmunications Program



- Easy Installation
- Menu Driven
- Intelligent computer to computer communications
- Supports most file transfer protocols
- Transfers CPM files to your system (Christensen Protocol)
- Access to timesharing services (Source, Compuserve)
- Available for OS/9 and Flex



Price: \$ 99.95

Great Plains Computer Company

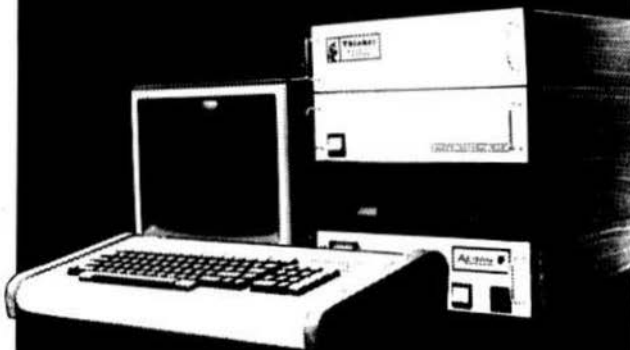
P. O. BOX 916
Idaho Falls, Idaho 83403
(208) 529-3210

OS/9 is a trademark of Microware

Flex is a trademark of TSC, Inc.

ACORN

COMPUTER SYSTEMS 88-50C



MODULES - BARE CARDS - KITS - ASSEMBLED & TESTED

Stackable Modules	KIT	A&T
20 AMP POWER SUPPLY w/fan w/Disk protect relay	350.00	400.00
DISK CABINET w/rege. & cables less DRIVES	200.00	250.00
MOTHER BOARD, 2 88-50c, 2 88-30c NMI button	225.00	325.00

Item	Bare	KIT	A&T
ITS - INTERRUPT TIMER 1, 10, 100 per sec. 19.95	29.95	39.95	
PB4 - INTELLIGENT PORT BUFFER Single board comput. 39.95	114.95	139.95	
DPIA - Dual PIA parallel port, 4 buffered I/Os 24.95	69.95	89.95	
XADE - Extended Addressing BAUD gen. PIA port 29.95	69.95	89.95	
MB8 - MOTHER BOARD 88-50c w/BAUD gen. 64.95	149.95	199.95	
P168 - 168K PROM DISK 21, 2764 EPROMs 39.95	79.95	109.95	
FD88 - Firmware development 2, 8K blocks 39.95	84.95	114.95	
XMR - 2764 PROM burner adapt. for 2716 BURNER 19.95	-----	-----	
CHERRY Keyboard w/Cabinet 96 key capacitive 249.95	-----	-----	
TAMM 12", 18 Mhz MONITOR GREEN AMBER 149.95	-----	159.95	
4 MODULE CABINET - unfinished POWER SUPPLY w/disk protect 250.00	-----	-----	

Color Computer

MONOLINK - 20 Mhz Monochrome video driver 15.00	20.00	
CC30 PORT BUS w/power supply 5 88-30, 2 Cart 169.95	199.95	
POWER BOX 6 switched outlets transient suppression 29.95	39.95	
RS-232 3-switched ports for above ADD +20.00	+25.00	

Write for FREE Catalog

ADD \$3.00 S&H PER ORDER
WIS. ADD 5% SALES TAX



11931 W. Bluemound Road
MILWAUKEE, WIS. 53226
(414) 257-0300

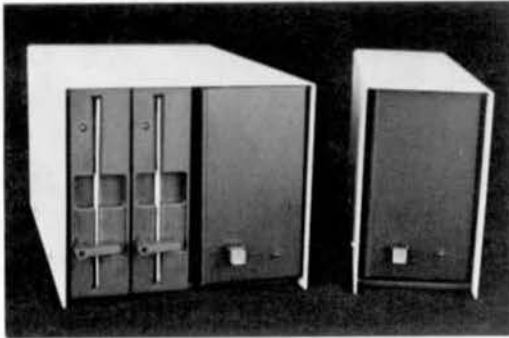
68' MICRO JOURNAL ADVERTISERS INDEX

'68' MICRO JOURNAL	67,68
ACORN COMPUTER SYSTEMS	70
ANDERSON COMPUTER CONSULTANTS	67
COMPILER EVALUATION SERVICES	59
COMPUTER EXCELLENCE INC.	60
COMPUTER PUBLISHING INC.	5
COMPUTER SYSTEMS CENTER	66
COMPUTER SYSTEMS CONSULTANTS, INC. ...	64
DATA-COMP	IBC
DIGITAL RESEARCH COMPUTERS	62,63
D.P. JOHNSON	59
EAPCO	69
GDMIX, INC.	3,72
GREAT PLAINS COMPUTER CO.	69
HAZELWOOD COMPUTER SYSTEMS	IBC
INTRIQ CORP.	61
LLOYD I/O	68
MICROWARE SYSTEMS CORP.	1,4
PERIPHERAL TECHNOLOGY	71
SARDIS TECHNOLOGIES	60
SNOKE SIGNAL BROADCASTING	6
SOUTH EAST MEDIA	35,36,37,38
SOUTHWEST TECHNICAL PRODUCTS INC. ...	IFC
STAK-KITS	67
TALBOT MICROSYSTEMS	64
TERMINUS DESIGN INC.	69
TMP SOFTWARE	60
UNITEX	68
WESTCHESTER APPLIED BUSINESS SYSTEMS	71
WINDRUSH MICRO SYSTEMS LIMITED	65

This index is provided as a reader service. The publisher does not assume any liability for omissions or errors.

PT-69 SINGLE BOARD COMPUTER SYSTEM OS-9 OR STAR-DOS NOW INCLUDED

- 1 MHz 6809E Processor
- 2 RS232 Ports (8850)
- 2 8-bit Ports (6821)
- 56K RAM 2K/4K EPROM
- Time-of-Day Clock
- 2797 Floppy Disk Controller



—PT68S2-40	Complete System with PT-68 Board, 2 DS/DD 5 1/4" 40 TR Drives, Cabinet, Power Supply. Your choice of OS-9 or STAR-DOS.	\$999.95
—PT-68S	Assembled & Tested Board with Power Supply and Cabinet.	\$399.95
—PT-69A	Assembled and Tested Board.	\$295.95
—Parallel Printer Interface with Cables		\$ 49.95
—OS-9 Level 1		\$200.00
—STAR-DOS Level 1		\$50.00

PERIPHERAL TECHNOLOGY

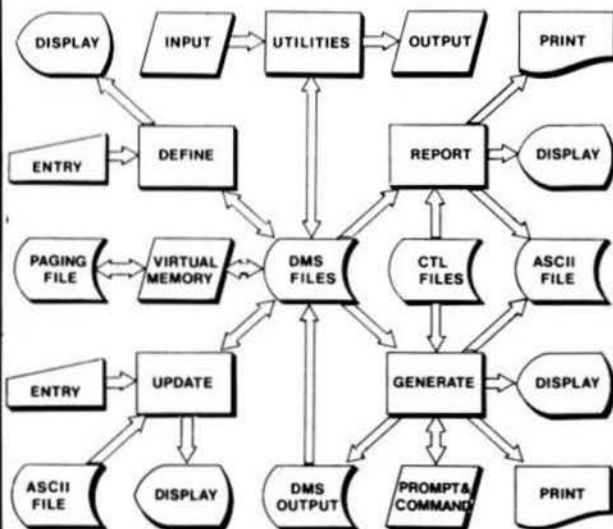
"Supplying Your Computer Needs Since 1978"

3760 Lower Roswell Road
Marietta, Georgia 30067

VISA/MASTERCARD/CHECK/COD 404/973-0042

XDMS

Data Management System



System Architecture

WESTCHESTER Applied Business Systems
Post Office Box 187
Briarcliff Manor, N.Y. 10510

XDMS Data Management System

The XDMS Data Management System is available in three levels. Each level includes the XDMS nucleus, VMOER utility and System Documentation for level III. XDMS is one of the most powerful systems available for 6809 computers and may be used for a wide variety of applications. XDMS users are registered in our database to permit distribution of product announcements and validation of user upgrades and maintenance requests.

XDMS Level I

XDMS Level I consists of DEFINE, UPDATE and REPORT facilities. This level is intended as an "entry level" system, and permits entry and reporting of data on a "tabular" basis. The REPORT facility supports record and field selection, field merge, sorting, line calculations, column totals and report titling. Control is via a English-like language which is upward compatible with level II. XDMS Level I \$129.95

XDMS Level II

Level II adds to Level I the powerful GENERATE facility. This facility can be thought of as a general file processor which can produce reports, forms and form letters as well as file output which may be re-input to the facility. GENERATE may be used in complex processing applications and is controlled by a English-like command language which encompasses that used by Level I. XDMS Level II \$199.95

XDMS Level III

Level III includes all of level II plus a set of useful DMS Utilities. These utilities are designed to aid in the development and maintenance of user applications and permit modification of XDMS system parameters, input and output of XDMS files, display and modification of file format, graphic display of numerical data and other functions. Level III is intended for advanced XDMS users. XDMS Level III \$299.95
XDMS System Documentation only \$10. credit toward purchase. . . \$ 24.95

XACC Accounting System

The XACC General Accounting System is designed for small business environments of up to 10,000 accounts and inventory items. The system integrates accounting functions and inventory plus the general ledger, accounts receivable and payable functions normally sold separately in other systems. Features user defined accounts, products for services, transactions, invoicing, etc. Easily configured to most environments. XACC General Accounting System (requires XDMS, Prof. Lv. III) . . \$299.95
XACC System Documentation only \$10. credit toward purchase. . . \$ 24.95

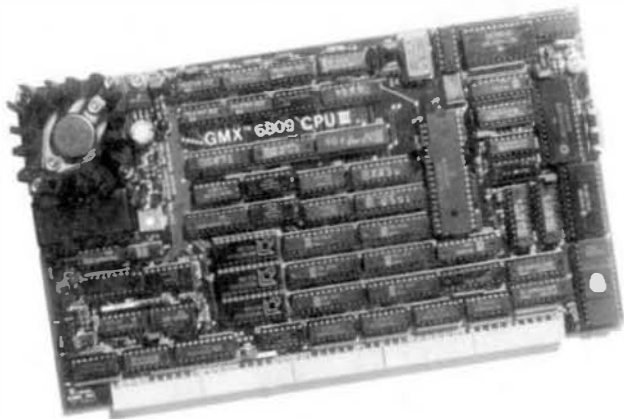
WESTCHESTER Applied Business Systems
Post Office Box 187, Briarcliff Manor, N.Y. 10510

All software is written in macro/assembler and runs under 6809 FLEX O/S. Terms: Check, Money Order, Visa or Mastercard. Shipment first class. Add P&H \$2.50 (\$7.50 Foreign). NY Res add sales tax. Specify 5" or 8".

Sales: S. E. MEDIA, 1-800-338-6800, Consultation: 914-941-3552 (evening).

FLEX is a trademark of Technical Systems Consultants, Inc.

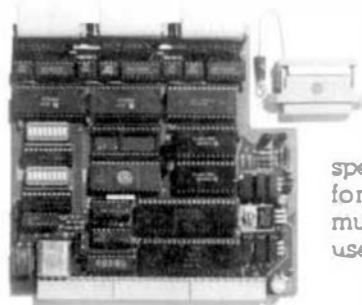
GIMIX STATE OF THE ART 6809 SYSTEMS FOR THE SERIOUS USER.



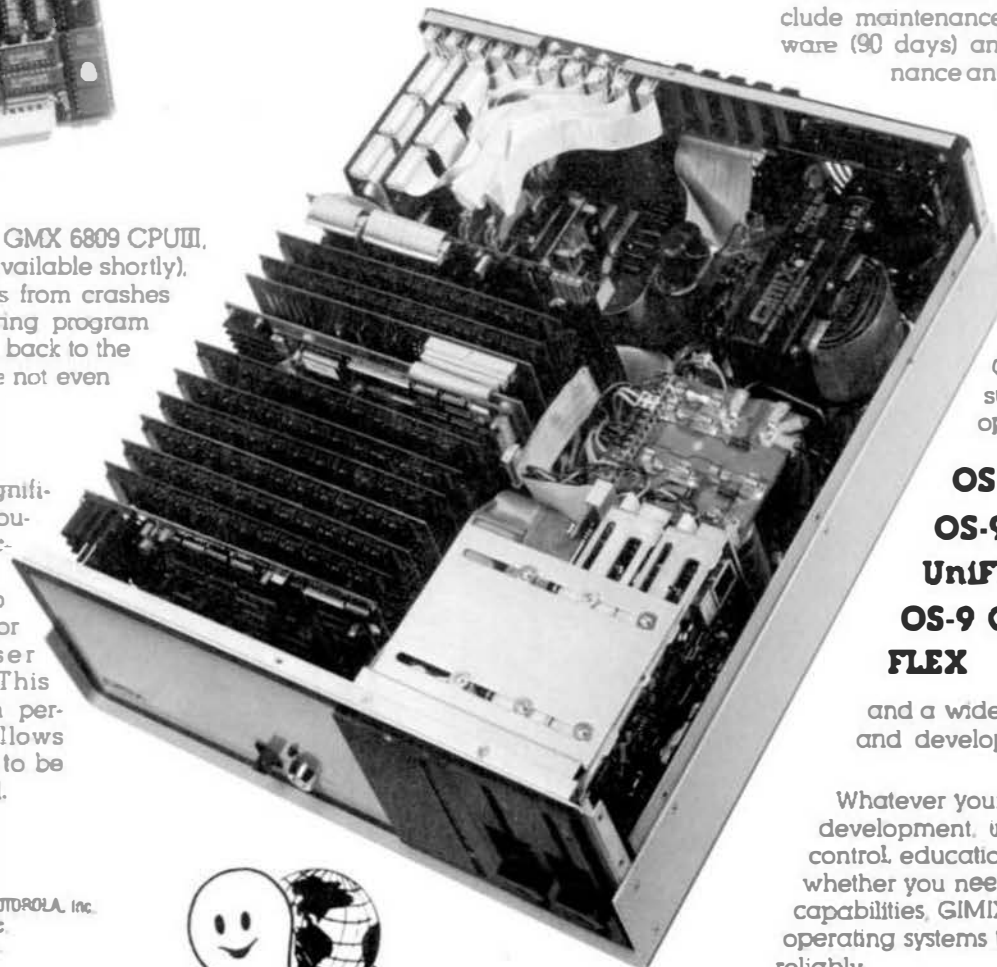
**GIMIX has 19MB or high performance
47MB Winchester Drive Systems and/or
Floppy Disk Drive Systems.**

For the ultimate in performance, the Unique GIMIX 6809 CPU, using either OS-9-GMX III or UniFLEX GMX III (available shortly), gives protection to the system and other users from crashes caused by defective user programs. e.g. During program development, a programmer who crashes goes back to the shell or the debugger, while the other users are not even aware anything occurred.

The intelligent serial I/O processor boards significantly reduce system overhead by handling routine I/O functions, thereby freeing up the host CPU for running user programs. This speeds up system performance and allows multiple terminals to be used at 19.2K baud.



BASIC-09 and OS-9 are trademarks of Microware Systems Corp. and MOTOROLA, Inc. FLEX and UniFLEX are trademarks of Technical Systems Consultants, Inc. GIMIX, GHOST, GMX, CLASSY CHASSIS, are trademarks of GIMIX, Inc.



GIMIX 6809 systems support five predominant operating systems:

**OS-9 GMX III,
OS-9 GMX II,
UniFLEX,
OS-9 GMX I,
FLEX**

and a wide variety of languages and development software.

Whatever your application, software development, instrumentation, process control, educational, scientific or business, whether you need single or multi-user capabilities, GIMIX has hardware and the operating systems to get the job done reliably.

Please phone or write if you need further information.



GIMIX inc.

1337 WEST 37th PLACE • CHICAGO, ILLINOIS 60609 • (312) 927-5510 • TWX 910-221-4055

© 1983 GIMIX Inc.



C.P.I.
Color Micro Journal
'68' Micro Journal
Data-Comp
S.E. Media

A Family of 100% 68XX Support Facilities
The Folks who FIRST Put FLEX™ on
The CoCo
Now Offering: *FLEX™ (2 Versions)
AND *STAR-DOS PLUS+™

FLEX-CoCo Sr.
with TSC Editor
TSC Assembler
Complete with Manuals
Reg. '250.00' **Only '79.00'**

STAR-DOS PLUS+
• Functions Same as FLEX
• Reads - writes FLEX Disks *34.00
• Run FLEX Programs
• Just type: Run "STAR-DOS"
• Over 300 utilities & programs
to choose from.

FLEX-CoCo Jr.
without TSC
Editor & Assembler
'49.00'

PLUS

ALL VERSIONS OF FLEX & STAR-DOS+ INCLUDE

TSC Editor
Reg \$50.00
NOW \$35.00

- + Read-Write-Dir RS Disk
- + Run RS Basic from Both
- + More Free Utilities
- + Super 800 Support
- + Free Color Micro Journal 1 yr. sub.

- + External Terminal Program
- + Test Disk Program
- + Disk Examine & Repair Program
- + Memory Examine Program
- + Many Many More!!!

TSC Assembler
Reg \$50.00
NOW \$35.00

CoCo Disk Drive Systems

NEW LOWER PRICES ON PAK #5. AND PRINTERS

THESE PACKAGES INCLUDE DRIVE, *CONTROLLER,
POWER SUPPLY & CABINET, CABLE, AND MANUAL.

* SPECIFY WHAT CONTROLLER YOU WANT J&M, OR RADIO SHACK.

PAK #1 - 1 SINGLE SIDED, DOUBLE DENSITY SYS.	\$349.95
PAK #2 - 2 SINGLE SIDED, DOUBLE DENSITY SYS.	\$639.95
PAK #3 - 1 DOUBLE SIDED, DOUBLE DENSITY SYS.	\$439.95
PAK #4 - 2 DOUBLE SIDED, DOUBLE DENSITY SYS.	\$699.95
PAK #5 - 2 DOUBLE SIDED, DOUBLE DENSITY SYS. THINLINE DRIVES, HALF SIZE	\$499.95

Controllers

J&M DISK CONTROLLER W/ JOOS OR RADIO SHACK
DISK BASIC, SPECIFY WHAT DISK BASIC. \$134.95

RADIO SHACK DISK CONTROLLER 1.1 \$134.95

Misc.

64K UPGRADE W/MOD. INSTRUCTIONS, C,D,E,F, AND COCO 2	\$ 44.95
HJL KEYBOARDS	\$ 74.95
MICRO TECH LOWER CASE ROM ADAPTER	\$ 74.95
RADIO SHACK BASIC 1.2	\$ 24.95
RADIO SHACK DISK BASIC 1.1	\$ 24.95
DISK DRIVE CABINET & POWER SUPPLY	\$ 49.95
SINGLE SIDED, DOUBLE DENSITY 5" DISK DRIVE	\$199.95
DOUBLE SIDED, DOUBLE DENSITY 5" DISK DRIVE	\$249.95

Printers

EPSON RX-80	\$269.00
EPSON RX-80FT	\$369.00
EPSON MX-100	\$499.00
EPSON FX-100	\$799.00
EPSON FX-80	\$549.00
EPSON MX-70	\$200.00

Disk Drive Cables

CABLE FOR ONE DRIVE	\$ 19.95
CABLE FOR TWO DRIVES	\$ 24.95

DATA-COMP

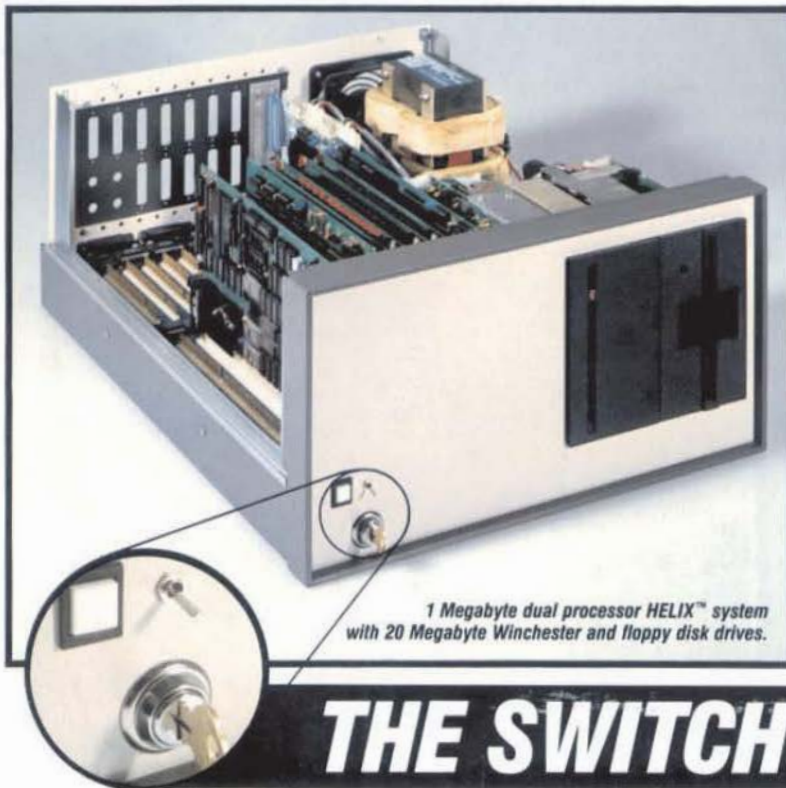
5900 Cassandra Smith Rd.
Hixson, TN 37343



SHIPPING
USA ADD 2%
FOREIGN ADD 5%
MIN. \$2.50

TOLL FREE
1-800-338-6800
For Ordering
TELEX 558 414 PVT BTH

KINGSTON SPRINGS IN 37082
P.O. BOX 87
MR. WICKY FERGUSON
000422 A/E



1 Megabyte dual processor HELIX™ system
with 20 Megabyte Winchester and floppy disk drives.

ONCE AGAIN HAZELWOOD COMPUTER SYSTEMS demonstrates its leadership in computer technology by delivering the only computer system capable of switching between either the 6809 or the 68000 processor. Switching is easily accomplished by a simple front panel toggle switch. The reason we can offer this exclusive feature now, is that when our proven 6809 processor board was designed several years ago, we had the foresight to include the bus controls that allow processor switching.

Hazelwood Computer Systems is also proud to be the first S-50/S-64 bus manufacturer to license and deliver the OS9/68K Operating System from Microware Systems Corporation. OS9/68K is the 68000 version of the popular and powerful OS9 Operating System. Utilizing our proven MC-20 disk controller, OS9/68K can conveniently share a Winchester disk with OS9. Changing from 6809 to 68000 operation is as simple as switching processors and booting the new system from the Winchester disk.

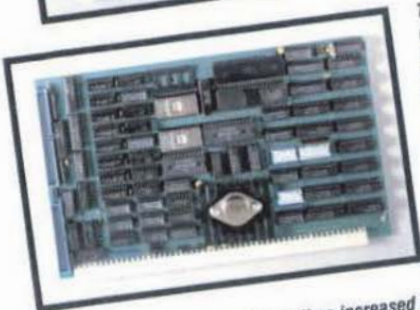
The ease of switching processors and operating systems makes a HELIX™ dual processor system the natural choice for software development. In addition, the advanced design of HELIX™ equipment, emphasizing performance and reliability, makes HELIX™ boards and systems the best value in computing offered anywhere.

System prices vary with configuration. Call for exact pricing.

THE SWITCH IS ON...



The CP-08 processor board utilizes a 68008 processor running at 10 Mhz clock rate. Using proprietary bus synchronization circuitry and single cycle DMA, the CP-08 achieves a marked performance increase over a 2 Mhz 6809. Offering absolute compatibility with the 68000 instruction set, the 68008 addresses up to 1 Megabyte of memory. Also included on the CP-08 are up to 4K of ROM, an interrupt timer, and with battery backup operation, a clock/calendar and 2K RAM. Implemented as a standard S-50 board, the CP-08 brings 68000 operation to S-50 bus computers.
PRICE: \$595
ORDER: CP-08



The MC-20 Mass Storage Controller board interfaces up to 4 floppy and 8 Winchester disk drives to the S-50/S-64 bus. The MC-20 is an intelligent controller with its own 2 Mhz 6809 processor and 56K RAM. It provides DMA data transfers to a full 24 bit address. All disk operation requests are by logical block number, with the controller performing the necessary track/sector address calculations. Any combination of 5 1/4 or 8 inch floppy drives can be accommodated with all drive parameters, such as write precompensation, software controlled for each individual drive. Winchester drives are connected via a SASI bus interface. Block address mapping is provided which allows a single drive to be segmented into several logical units. The MC-20 is the controller of the MS-20 Mass Storage Subsystem which includes a 20 Megabyte Winchester drive.
PRICE: \$695
ORDER: MC-20

OS9/68K offers increased performance and larger user memory space while retaining all of the features of OS9. Disk file compatibility and operational similarity assures that present OS9 users can easily transfer their operations to the 68000. Included are an editor, assembler, linker, and debugger. A C compiler is available now. BASIC09 and other languages will be available soon.

OS9/68K

ORDER: OS9/68K

PRICE: \$250

All items available stock to 30 days.
Prices subject to change without notice.

HAZELWOOD COMPUTER SYSTEMS

907 East Terra, O'Fallon, MO 63366,

314-281-1055

OS9 and OS9/68K are registered trademarks of microware Systems Corp. HELIX is a trademark of Hazelwood Computer Systems.

HELIX